1-1-2015

Multi-Robot Active Information Gathering Using Random Finite Sets

Philip Dames
University of Pennsylvania, pdames@seas.upenn.edu

Multi-Robot Active Information Gathering Using Random Finite Sets

Abstract
Many tasks in the modern world involve collecting information, such as infrastructure inspection, security and surveillance, environmental monitoring, and search and rescue. All of these tasks involve searching an environment to detect, localize, and track objects of interest, such as damage to roadways, suspicious packages, plant species, or victims of a natural disaster. In any of these tasks the number of objects of interest is often not known at the onset of exploration. Teams of robots can automate these often dull, dirty, or dangerous tasks to decrease costs and improve speed and safety. This dissertation addresses the problem of automating data collection processes, so that a team of mobile sensor platforms is able to explore an environment to determine the number of objects of interest and their locations. In real-world scenarios, robots may fail to detect objects within the field of view, receive false positive measurements to clutter objects, and be unable to disambiguate true objects. This makes data association, i.e., matching individual measurements to targets, difficult. To account for this, we utilize filtering algorithms based on random finite sets to simultaneously estimate the number of objects and their locations within the environment without the need to explicitly consider data association. Using the resulting estimates they receive, robots choose actions that maximize the mutual information between the set of targets and the binary events of receiving no detections. This effectively hedges against uninformative actions and leads to a closed form equation to compute mutual information, allowing the robot team to plan over a long time horizon. The robots either communicate with a central agent, which performs the estimation and control computations, or act in a decentralized manner. Our extensive hardware and simulated experiments validate the unified estimation and control framework, using robots with a wide variety of mobility and sensing capabilities to showcase the broad applicability of the framework.

Degree Type
Dissertation

Degree Name
Doctor of Philosophy (PhD)

Graduate Group
Mechanical Engineering & Applied Mechanics

First Advisor
Vijay Kumar

Subject Categories
Computer Sciences | Mechanical Engineering | Robotics

MULTI-ROBOT ACTIVE INFORMATION GATHERING USING RANDOM FINITE SETS

Philip M. Dames

A DISSERTATION

in

Mechanical Engineering and Applied Mechanics

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

2015

Vijay Kumar, Supervisor of Dissertation
UPS Foundation Professor of Mechanical Engineering and Applied Mechanics

---

Prashant Purohit, Graduate Group Chairperson
Associate Professor of Mechanical Engineering and Applied Mechanics

Dissertation Committee

George Pappas, Professor of Electrical and Systems Engineering
Vijay Kumar, Professor of Mechanical Engineering and Applied Mechanics
Daniel E. Koditschek, Professor of Electrical and Systems Engineering
Mac Schwager, Assistant Professor of Mechanical Engineering

MULTI-ROBOT ACTIVE INFORMATION GATHERING USING RANDOM FINITE SETS

Philip Mayotte Dames

Dedicated to Lily

Acknowledgments

I would like to thank my advisor, Vijay Kumar, whose has guided and supported me over the last five years. I sincerely appreciate all of the opportunities that he has provided for me to become an independent scholar, to meet interesting people, and to travel across the globe. I would also like to thank the rest of my dissertation committee, George Pappas, Dan Koditschek, and Mac Schwager, for taking the time to serve on my committee and for the interesting discussions over the past several years. Mac Schwager, in particular, helped me build a solid foundation for my research during his tenure as a postdoctoral fellow as Penn.

Kevin Lynch, Paul Umbanhowar, and Tom Vose first got me interested in the field of robotics through their mentorship during my undergraduate years. They really challenged me to do independent work and gave me a head start for when I arrived at Penn.

I would like to thank all of the current and former members of GRASP that I have had the pleasure to work with. In particular I would like to thank Ben Charrow for helping turn a mechanical engineer into a passable computer scientist and for the many long discussions about probability, information theory, and life. I would also like to thank Justin Thomas for helping me finally learn how to use quadrotors during the last month of my dissertation.

I would never have gotten here without the constant support from my family. My parents provided me with every opportunity to succeed in life and I can't thank them enough.

Finally, I would like to thank my wife, Lily, who always reminds me that there is a great big world full of adventure (and music) outside of the laboratory. She has also been instrumental in helping make this dissertation, and many of my papers, more organized and readable, even without knowing all of the technical jargon filling them.

ABSTRACT

MULTI-ROBOT ACTIVE INFORMATION GATHERING USING RANDOM FINITE SETS

Philip M. Dames

Vijay Kumar

Many tasks in the modern world involve collecting information, such as infrastructure inspection, security and surveillance, environmental monitoring, and search and rescue. All of these tasks involve searching an environment to detect, localize, and track objects of interest, such as damage to roadways, suspicious packages, plant species, or victims of a natural disaster. In any of these tasks the number of objects of interest is often not known at the onset of exploration. Teams of robots can automate these often dull, dirty, or dangerous tasks to decrease costs and improve speed and safety. This dissertation addresses the problem of automating data collection processes, so that a team of mobile sensor platforms is able to explore an environment to determine the number of objects of interest and their locations. In real-world scenarios, robots may fail to detect objects within the field of view, receive false positive measurements to clutter objects, and be unable to disambiguate true objects. This makes data association, *i.e.*, matching individual measurements to targets, difficult. To account for this, we utilize filtering algorithms based on random finite sets to simultaneously estimate the number of objects and their locations within the environment without the need to explicitly consider data association. Using the resulting estimates they receive, robots choose actions that maximize the mutual information between the set of targets and the binary events of receiving no detections. This effectively hedges against uninformative actions and leads to a closed form equation to compute mutual information, allowing the robot team to plan over a long time horizon. The robots either communicate with a central agent, which performs the estimation and control computations, or act in a decentralized manner. Our extensive hardware and simulated experiments validate the unified estimation and control framework, using robots with a wide variety of mobility and sensing capabilities to showcase the broad applicability of the framework.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Mobile computers, sensors, and robot platforms are becoming more powerful and less expensive, particularly as smartphones, wearable devices, and hobby robots become mainstream. These technologies can be combined to create low cost mobile sensor platforms. While individual robots built from low-cost components have limited capabilities, putting multiple robots together into a team increases their collective computational power, the effective sensor field of view, and the robustness of the team to individual agent or sensor failure. As the technologies continue to mature, teams of robots will automate more tasks that are dull, dirty, dangerous, or not possible for humans to perform.

Such autonomy requires robots to sense the surrounding environment, to communicate with other robots, to make reasoned decisions about the environment, and to select actions that will quickly lead to completing the mission at hand. This dissertation focuses on a broad class of problems related to information gathering. In information gathering tasks, a robot team begins with an incomplete understanding of the surrounding environment, and the task is to improve this understanding to some desired level. Examples of information gathering tasks include security and surveillance, where the robots search a known environment for intruders, damage, or suspicious activity; environmental monitoring, where the robots search an area for specific chemical signals, species of flora or fauna, or crop health; mapping, where the robots search a known area to map out specific features or explore a new area in order

Table 1: Example information gathering applications.

Scenario	Targets	Number of targets?	Targets moving?
Infrastructure inspection	Damage or wear	Small	No
Security and surveillance	Intruders	Small	Sometimes
Map registration	Smart devices	Large	No
Precision agriculture	Crop health	Large	No
Environmental monitoring	Plant species	Large	Sometimes
Reconnaissance	Enemy assets	Large	Sometimes
Search and rescue	People	Large	Sometimes

to build a map; and search and rescue, where the robots seek out lost or injured individuals. Such tasks span many geographic and temporal scales: search and rescue missions are often confined to a small area and must be completed in a manner of minutes while environmental monitoring missions may take place over many kilometers and last months or years. Table 1 details these information gathering scenarios.

All of these tasks share the same high-level goal: to identify the locations of all of the objects of interest *e.g.*, intruders or map landmarks, within the environment. However, the number of objects of interest is often not known at the onset of exploration, and the objects may not be uniquely identifiable. For example, in an environmental monitoring task two plants of the same species may look identical. Additionally, the sensors on board the robot may be unreliable: failing to detect objects within their field of view, providing false positive measurements, and providing noisy measurements of true objects. The number of objects within the sensor field of view may also change over time, due to motion of the robots, motion of the objects, or obstacles in the environment. It is important for any perception and decision making framework to take these uncertainties into account when estimating the state of the surrounding environment and selecting actions to improve this estimate.

The multi-target tracking problem has received considerable attention from many fields of study, most prominently from the tracking and simultaneous localization and mapping (SLAM) communities. Many approaches use approximations or heuristics to apply classical estimation algorithms that assume the number of objects and the data association, *i.e.*, the matching of measurements to targets, are known. Other approaches develop novel estimation

algorithms to account for the uncertainties in the environment and sensor readings. While the former group of estimation algorithms has received some attention from the active sensing community, the latter has not. Chapter 2 provides a survey of tracking algorithms, a brief tutorial on the methods used in this dissertation, and an overview of active sensing methods.

This dissertation contributes to these developing technologies by enabling teams of mobile robots to autonomously explore and gather information with limited *a priori* knowledge of the given situation, turning sensor data into actionable information. In any of these information gathering scenarios there may be uncertainty in the environment, the number of objects of interest may be unknown, or there may be unpredictable physical phenomena. This research aims to improve the performance of robotic teams in real-world application domains by building systems that explicitly consider such uncertainties. The closed form control objective developed in this dissertation accounts for these uncertainties while allowing a small team of robots to jointly plan actions over a finite horizon in real time. Robots working together as a team are able to gather information more quickly and efficiently than robots exploring independently. However, such coordination is not possible in many situations due to limitations in wireless communication. The proposed framework is flexible, allowing the team the use either a central planner when possible or decentralized coalitions that are formed online when communication is limited.

We apply our unified estimation, control, and communication framework to a variety of information gathering problems. Chapter 3 begins with the simplest problem, where the number of objects of interest is not known precisely, but is believed to be small. This fits naturally with security and surveillance or infrastructure inspection situations, where there are often no intruders or damaged areas. In such scenarios it is also reasonable to assume that the robot team has a prior map of the environment and that the robots are able to localize themselves within this map. Chapter 4 expands the estimation and control framework to actively seek out a large number of objects of interest within a known search space. This has applications to precision agriculture and map registration. Finally, Chapter 5 extends the framework to detect, localize, and track a large number of moving objects, with applications

to environmental monitoring and surveillance.

We test and verify our proposed framework through simulated and real-world experiments, using robots with a variety of mobility and sensing capabilities to demonstrate the flexibility and broad applicability of the framework. Chapter 3 presents experiments with a large, differential drive ground robot equipped with a monocular camera as well as with a small team of quadrotor Micro Aerial Vehicles (MAVs) equipped with magnetometer sensors. Chapter 4 presents results using the Scarab platform, a small, differential-drive robot designed and built here at the University of Pennsylvania, exploring an indoor office environment. Finally, Chapter 5 presents simulation results using fixed-wing aircraft equipped with downward-facing cameras, using a real-world data set for target motion. To the best of our knowledge, the results in this dissertation are the first multi-robot and experimental results of an active exploration strategy based on random finite sets.

Chapter 2

Background Material

This chapter reviews many of the multi-object estimation and multi-robot control concepts that will be used throughout the dissertation. Section 2.1 reviews concepts in single- and multi-target tracking and justifies our selection of tracking algorithms. Section 2.2 provides a tutorial on finite set statistics, the mathematical tool used in our estimation algorithm. Finally, Section 2.3 provides an overview of active information gathering approaches and positions our framework with the current state of scholarship and research on the subject.

2.1 Target Tracking

Target tracking is a broad class of problems, with applications in aerospace systems, image processing, oceanography, remote sensing, biomedical research, and robotics. These applications include everything from tracking objects flying through a specific region of airspace to feature-based mapping of an unknown environment. While there are many different ways to classify tracking problems, the most important factors are whether the number of targets is known, whether the data association is known, and whether the estimation is performed sequentially or is processed as one batch. Data association is the process of matching measurements to target tracks. This is a critical component of any tracking problem, as incorporating incorrect evidence into the estimate of a target can cause the uncertainty to grow or, in the worst case, can cause the estimate to diverge.

This section reviews common methods for Bayesian single-target tracking, namely the Kalman filter, the histogram filter, and the particle filter. We will then discuss existing methods to extend these single-target approaches to situations with multiple targets. Finally we present Bayesian methods for multi-target tracking, focusing on approaches where the number of targets and the data association are both unknown.

2.1.1 Single-Target Tracking

Single-target tracking is a canonical problem in estimation theory and robotics. In the single-target scenario, the data association is known since there is only a single target. Thus, the problem is simply how to use the incoming measurements from the sensors to update the belief about the state of the target. Let E be the environment that the robot team explores and let $\mathbf{q} \in E$ be the pose of the robot. Let $\mathbf{x} \in E$ be the state of the target and let $\mathbf{z}$ be a measurement from a robot's sensor. Since there is uncertainty associated with each of these quantities, we will use a probabilistic representation.

It is possible for some targets to move over time. To account for this possibility, we define the motion model $f(\mathbf{x} \mid \xi)$, which defines the probability of a target with initial state ξ moving to state $\mathbf{x}$. In situations with stationary targets the transition model is simply the identity map, $f(\mathbf{x} \mid \xi) = \delta_\xi(\mathbf{x})$, where $\delta_\xi(\mathbf{x})$ is the Kronecker delta function.

Let $g(\mathbf{z} \mid \mathbf{x}, \mathbf{q})$ be the probability of a robot with pose $\mathbf{q}$ receiving a measurement $\mathbf{z}$ from a target with state $\mathbf{x}$. Let $p(\mathbf{x})$ be the prior probability that the target has state $\mathbf{x}$. We may then use Bayes' rule to find the posterior probability,

$$p(\mathbf{x} \mid \mathbf{z}) = \frac{p(\mathbf{x}, \mathbf{z})}{p(\mathbf{z})} = \frac{p(\mathbf{z} \mid \mathbf{x})p(\mathbf{x})}{\int p(\mathbf{x}, \mathbf{z})\, d\mathbf{x}} = \frac{g(\mathbf{z} \mid \mathbf{x}, q)p(\mathbf{x})}{\int g(\mathbf{z} \mid \mathbf{x}, \mathbf{q})p(\mathbf{x})\, d\mathbf{x}}. \tag{2.1}$$

This is known as the Bayes filter and is the most general sequential estimation algorithm. However, it is not possible to maintain an arbitrary distribution over all possible target states using this method. In order for the Bayes filter to be computationally tractable, we must make some simplifying assumptions.

Kalman Filter

The Kalman filter (KF) [53] is an implementation of the Bayes filter for linear Gaussian systems. This means that the target state is represented by a Gaussian distribution, the measurement and motion models are linear, and all noise is additive Gaussian. The Gaussian distribution takes the form

$$p(\mathbf{x}) = \det(2\pi\Sigma)^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right), \tag{2.2}$$

where μ is the mean of the distribution, Σ is the covariance matrix, $\det(\cdot)$ is the determinant of a matrix, and $\mathbf{x}$ is a vector of the target state. Figure 1a shows an example distribution. Note that this distribution is fully characterized by the mean vector and covariance matrix. The Kalman filter provides a set of rules to update these parameters.

Let the transition model be

$$\mathbf{x}_{t|t-1} = A_t\mathbf{x}_{t-1} + \mathbf{b}_t + \epsilon_t, \tag{2.3}$$

where A_t is a matrix, $\mathbf{x}_{t-1}$ is the prior state at time $t-1$, $\mathbf{x}_{t|t-1}$ is the predicted state, $\mathbf{b}_t$ is an affine term (often representing the control input at time t), and ϵ_t is a Gaussian random vector with zero mean and covariance R_t. Then the update equations are

$$\mu_{t|t-1} = A_t\mu_{t-1} + \mathbf{b}_t \tag{2.4}$$

$$\Sigma_{t|t-1} = A_t\Sigma_{t-1}A_t^T + R_t. \tag{2.5}$$

Let the measurement model be

$$\mathbf{z} = C_t\mathbf{x}_t + \delta_t, \tag{2.6}$$

where C_t is a matrix and δ_t is a Gaussian random vector with mean zero and covariance Q_t.

Then the update equations are

$$K_t = \Sigma_{t|t-1} C_t^T (C_t \Sigma_{t|t-1} C_t^T + Q_t)^{-1} \tag{2.7}$$

$$\mu_t = \mu_{t|t-1} + K_t(\mathbf{z}_t - C_t \mu_{t|t-1}) \tag{2.8}$$

$$\Sigma_t = (I - K_t C_t)\Sigma_{t|t-1}, \tag{2.9}$$

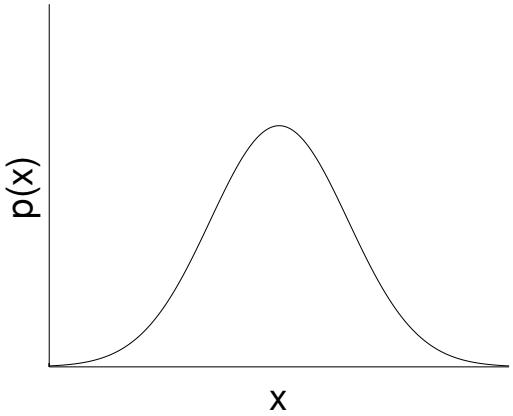
where I is the identity matrix and K_t is the so-called Kalman gain. Intuitively, the Kalman gain describes how much to trust the measured versus the predicted target location. The term $\mathbf{z}_t - C_t \mu_{t|t-1}$ is known as the innovation of the observation, and represents the difference between the predicted and actual measurement.

The Kalman filter can be extended to deal with non-linear measurement and motion models by linearizing about the mean, leading to the extended Kalman filter (EKF) [97, Chapter 3.3], or by using the unscented transform, leading to the unscented Kalman filter (UKF) [97, Chapter 3.4].

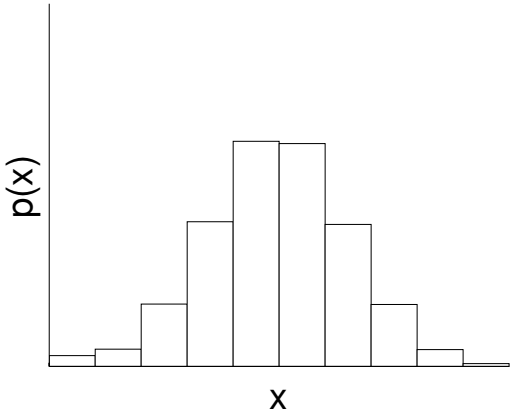
Histogram Filter

While the Kalman filter is computationally efficient and works well in some scenarios, it has a number of shortcomings. Namely, it can accumulate significant error when the measurement model is highly non-linear. Also the probabilistic representation is inherently unimodal, meaning it cannot be used to represent distributions with multiple, disjoint hypotheses. In some situations, particularly with non-linear measurements such as range-only measurements, multimodal distributions are common.

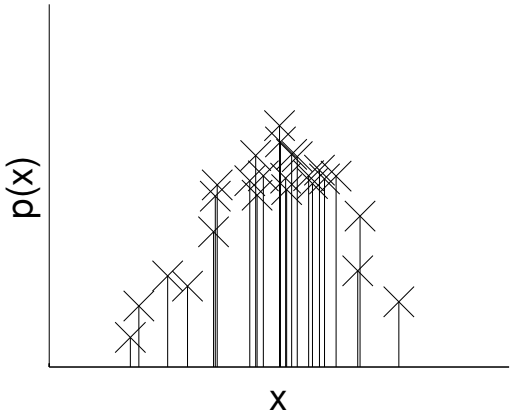
The histogram filter [97, Chapter 4.1] provides an alternative, non-parametric solution. First, the state space of the target is divided into a finite collection of regions. The histogram filter then provides a set of equations to update the probability that the target's state falls into each region. These regions are often of a uniform size, *e.g.*, a uniform grid, though this is not required. Figure 1b shows an example set of bins that approximates the distribution in Figure 1a.



(a) Kalman filter



(b) Histogram filter



(c) Particle filter

Figure 1: Example unimodal distribution represented by (a) a Gaussian distribution, (b) a histogram filter, and (c) a particle filter.

The prediction and update rules are then quite simple,

$$p_{k,t|t-1} = \sum_i f(\mathbf{x}_{k,t} \mid \mathbf{x}_{i,t-1}) p_{i,t-1} \tag{2.10}$$

$$\eta = \sum_k g(\mathbf{z}_t \mid \mathbf{x}_k) p_{k,t|t-1} \tag{2.11}$$

$$p_{k,t} = \eta^{-1} g(\mathbf{z}_t \mid \mathbf{x}_k) p_{k,t|t-1} \tag{2.12}$$

where $p_{k,t}$ is the probability that the target is in region k at time t (*i.e.*, $\mathbf{x}_t \in x_{k,t}$) and η is a normalization constant. The accuracy and computational complexity of the histogram filter will depend upon the size of the histogram bins.

Particle Filter

The particle filter (PF) [97, Chapter 4.3] provides an alternative to the histogram filter. In this case the arbitrary distribution is represented by a set of weighted particles, rather than a set of cells. These particles may take arbitrary locations, though it is desirable to have more particles in areas with higher likelihood to better capture the behavior of the distribution. Let the number of particles be N, where $\mathbf{x}_{i,t}$ is the ith particle state (*i.e.*, hypothesis) and $w_{i,t}$ is the weight (*i.e.*, likelihood) of the particle. Figure 1c shows an example set of particles that approximates the distribution in Figure 1a.

In the most basic implementation of the PF, each particle moves according to the motion model, *i.e.*, $\mathbf{x}_{i,t|t-1} \sim f(\cdot \mid \mathbf{x}_{i,t-1})$. The weight of the particle is then updated

$$\eta = \sum_k g(\mathbf{z}_t \mid \mathbf{x}_{k,t|t-1}) w_{k,t|t-1} \tag{2.13}$$

$$w_{k,t} = \eta^{-1} g(\mathbf{z}_t \mid \mathbf{x}_{k,t|t-1}) w_{k,t|t-1}. \tag{2.14}$$

There are many variants of the particle filter, which involve periodically resampling the particles to remove low-likelihood particles and replace them with particles in areas of interest. Additionally, the number of particles may be scaled online, using more particles to represent distributions with higher uncertainty.

2.1.2 Multi-Target Tracking

The multi-target tracking problem is more complicated. The simplest case is when the number of targets and the data association are both known. In this case, a separate single-target filter may be used to track the states of the different targets $\mathbf{x}_i$. Each measurement is then used to update the associated target estimate. This is an intuitive and appealing approach, and much of the work on multi-target tracking attempts to reformulate the problem as a collection of single-target problems.

Having known data association is equivalent to stating that the sensor is able to uniquely identify individual targets and that it is able to do so without error. This is a valid assumption in some settings, *e.g.*, localizing wireless sensors using the MAC address to provide a unique label [12]. However, in many other systems there could easily be errors in the data association, *e.g.*, when tracking people in a crowd using facial recognition software, or association may not be possible, *e.g.*, when mapping a set of identical-looking doors in an office environment. In these cases, we must solve both the data association and tracking problems.

One common method of multi-target tracking with unknown associations is to use the maximum likelihood association. This has been used successfully in a variety of situations, including simultaneous localization and mapping [31]. In this approach, each measurement is checked against each object. Any measurement that is sufficiently close to the estimated target state is accepted as an association, ensuring that only one measurement is associated with each object. All other measurements are discarded or are used to initialize new target estimates. While this approach is simple to understand and implement, all of the associations decisions are "hard," meaning that there is no notion of uncertainty. This ideas runs counter to the probabilistic approach often used in tracking problems, and means making an incorrect association can have a long-lasting impact on the estimate. Mullane et al. [76] show examples of such errors in the context of feature-based mapping.

Another approach is the Multiple Hypothesis Tracker (MHT) [95, Chapter 4.2], which assumes the data association to be an unknown value. The MHT algorithm provides a

joint target state distribution conditioned on each data association. Let $\theta : \{1, \ldots, n\} \to \{0, 1, \ldots, m\}$ be an association between a set of n targets and a set of m measurements. Note that $\theta(j) = 0$ means that the target is not detected, *i.e.*, a false negative, and any element of $\{1, \ldots, m\}$ not in the range of $\theta(\{1, \ldots, n\})$ is a false positive. Then the update step, for a given association, becomes

$$p_t(\mathbf{x} \mid \mathbf{z}, \theta) = \eta^{-1} g(\mathbf{z} \mid \mathbf{x}, \theta) p_{t|t-1}(\mathbf{x}). \tag{2.15}$$

However, the number of associations grows combinatorially in the number of targets and measurements, making this approach intractable for large problems or long time horizons. The explosive growth in the number of association histories over time can be reduced, for example, by keeping the N best associations at each time step or the N best association histories. However, this still requires computing all possible associations at each time step.

The Probabilistic MHT (PMHT) method [95, Chapter 4.6] relaxes the MHT assumptions to allow for soft associations and to allow more than one measurement to be associated with a single target. It also assumes that the single-target measurement likelihoods are conditionally independent given a data association. PMHT then uses the expectation maximization (EM) algorithm to find the maximum a posteriori (MAP) estimate of the targets given the measurements. In this case, the association likelihoods and the number of targets are assumed to be known and fixed. PMHT is a batch method, solving for the sequence of maximum likelihood target tracks using a sequence of measurement sets. This makes it unsuitable for real-time applications.

Another canonical approach is the Joint Probabilistic Data Association (JPDA) [95, Chapter 4.5]. JPDA provides a set of single target estimates that are each a mixture of the posterior distributions resulting from each possible data association. This also assumes that the single-target measurement likelihoods are conditionally independent given a data

association. In this approach, we compute the soft association matrix A, with elements

$$a_{mn} = \sum_{\theta|\theta(m)=n} p(\theta \mid \mathbf{z}) \tag{2.16}$$

and let $\bar{a}_n = 1 - \sum_m a_{mn}$ be the probability that no measurement is associated to target n. Then the update rule for target n JPDA is

$$p_t(\mathbf{x}_n) = \bar{a}_n p_{t|t-1}(\mathbf{x}_n) + \sum_m a_{mn} \eta^{-1} g(\mathbf{z}_m \mid \mathbf{x}_n) p_{t|t-1}(\mathbf{x}_n). \tag{2.17}$$

The JPDA method can be extended to allow for new targets to enter the environment and for existing targets to disappear. While JPDA does treat data association as an unknown quantity, it assumes that the number of targets is known.

The final approach involves the use of intensity filters [95, Chapter 5], which do not explicitly compute associations or estimate target tracks. Instead, intensity filters compute intensity functions that estimate the density of targets in the target state space. The following section outlines the mathematics behind the intensity filter-based approaches and presents several of the estimation algorithms.

2.2 Finite Set Statistics

This section provides a summary of a set of tools known as Finite Set Statistics (FISST), which were first developed by Mahler [70]. The key feature of FISST is that collections of objects are represented by sets. This departs from traditional robotics solutions that use stacked vectors to represent collections of objects. Section 2.2.1 covers some key mathematical properties of sets and the resulting advantages in multi-target tracking, mapping, and other robotics applications. Section 2.2.2 summarizes the fundamental concepts of FISST. Finally, Section 2.2.3 provides a summary of filtering methods based on the FISST framework.

2.2.1 Vector- vs. Set-Based Representations

To more clearly highlight the differences between vector- and set-based representations, we will consider the task of feature-based mapping. This subsumes the multi-target tracking problem when the targets are stationary. In such problems the number of features within an environment is typically not known *a priori* and must be discovered online as the robot explores. Two key issues arise is such a setting:

- Feature management – tracking the landmarks within a map
- Data association – matching measurements to landmarks

Both of these issues are further complicated when there is uncertainty in the sensor. A robot may fail to detect a landmark that is present (a false negative measurement), may receive a spurious measurement (a false positive, or clutter, measurement), or may receive a noisy measurement to a true landmark.

Feature Management

Feature management is a difficult task when a robot explores an unknown environment. Vector-based approaches often follow the approach outlined by Dissanayake et al. [31], where the map is initialized as an empty vector. As the robot receives measurements, it initializes new features and appends them to the vector of map landmarks using heuristic rules. While such vector-based approaches have been successfully applied in many scenarios, they have several issues:

Issue 1. If a robot explores the same environment along two distinct routes then the landmarks may be added to the map state vector in a different order. Thus, the same environment has many possible representations, and naïve methods for comparing two maps (*e.g.*, the vector norm) may lead the user to conclude that two maps are radically different when they are actually the same.

Issue 2. There is no notion of uncertainty in a landmark's existence. Landmarks either exist or they do not and are added (or removed) using heuristic rules. This stems from the

fact that the number of landmarks is not treated probabilistically, only the locations of the landmarks are.

Issue 3. The dimensionality of the state space of the map changes over time as new features are discovered and added to the map. This makes it difficult to compare map estimates at different times, when the number of landmarks may have changed.

Issue 4. The dimensionality of the landmark state space is not immediately evident in the representation. For example, a vector with six elements may represent six one-dimensional landmarks, three two-dimensional landmarks, two three-dimensional landmarks, or one six-dimensional landmark.

A set-based approach instead represents the map as a set of landmarks, where the size of the set is the number of landmarks and individual elements in the set represent the states of the individual landmarks. Two mathematical properties of sets also solve all of the above issues with the vector-based approach. First, sets are equivalent under a permutation of the elements. This completely eliminates Issue 1, as it does not matter the order that landmarks are added to the map feature set. Second, sets have well-defined union and complement operators. These naturally handle the addition and removal of elements to the set. Issues 2 and 3 are resolved by tracking a distribution over feature sets. The distribution over the cardinality of the set explicitly tracks the belief in the number of targets and makes it possible for features to have a probability of existence. Finally, Issue 4 is solved by simply examining an individual element in the set, as each element represents an individual feature.

Data Association

Data association is a computationally challenging task in a mapping setting, as the number of possible associations grows combinatorially in the number of map features and the number of measurements. As Section 2.1 outlines, data association is often solved as a preprocessing step to landmark estimation using heuristics [31] or maximum likelihood estimates. However, with a vector-based approach, even in settings where there is no ambiguity in association, the vector of measurements may need to be permuted in order to apply the landmark updates.

This is due to the explicit ordering of elements within a vector. Conversely, if measurements are also represented as a set then data association is no longer an issue, as all of the different permutations are implicitly encoded in the sets.

We treat the data association as an unknown variable and remove the dependence on the association by marginalization. Consider the problem of data association between a set of n object $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and a set of m measurements $Z = \{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$. For the purposes of this example, we make the following assumptions:

A1. Each object generates a single detection with probability $p_d(\mathbf{x} \mid \mathbf{q})$ or zero detections with probability $1 - p_d(\mathbf{x} \mid \mathbf{q})$.

A2. The number of clutter objects follows a Poisson distribution with mean μ.

A3. The clutter measurements are independently and identically distributed (i.i.d.) with distribution $c(\mathbf{z} \mid \mathbf{q})$.

A4. The clutter measurements are conditionally independent of the true detections given the target states.

A5. Any two measurements in Z are conditionally independent given the target states.

Assumptions A1, A3, A4, and A5 are all standard for multi-target tracking problems. Justification for assumption A2 appears in Section 2.2.3. Without loss of generality, let us assume that all objects are detectable, $p_d(\mathbf{x} \mid \mathbf{q}) > 0, \ \forall \mathbf{x} \in X$. We will examine the data association problem in several cases.

Perfect sensor In this case there are no clutter detections and no false negative detections since $p_d(\mathbf{x} \mid \mathbf{q}) = 1, \ \forall \mathbf{x} \in X$. This means that the number of measurements must match the number of objects, or $m = n$, and the only valid associations θ are permutations of $\{1, \ldots, n\}$. The likelihood of the measurement set is then

$$p(Z \mid X, \mathbf{q}) = \sum_{\theta} \prod_{i=1}^{n} g(\mathbf{z}_{\theta(i)} \mid \mathbf{x}_i, \mathbf{q}). \tag{2.18}$$

No missed detections but clutter possible In this case $p_d(\mathbf{x} \mid \mathbf{q}) = 1,\ \forall \mathbf{x} \in X$, so $m \geq n$ and all valid associations are one-to-one mappings from $\{1, \ldots, n\} \rightarrow \{1, \ldots, m\}$. If the object set $X = \varnothing$, so all measurements are clutter, then

$$p(Z \mid \varnothing, \mathbf{q}) = e^{-\mu} \prod_{\mathbf{z} \in Z} c(\mathbf{z} \mid \mathbf{q}). \tag{2.19}$$

If $X \neq \varnothing$ then the likelihood is

$$p(Z \mid X, \mathbf{q}) = \sum_{\theta} \prod_{i=1}^{n} g(\mathbf{z}_{\theta(i)} \mid \mathbf{x}_i, \mathbf{q}) \prod_{\substack{j \mid \theta(i) \neq j \\ \forall i \in \{1, \ldots, n\}}} c(\mathbf{z}_j \mid \mathbf{q}) \tag{2.20}$$

$$= p(Z \mid \varnothing, \mathbf{q}) \sum_{\theta} \prod_{i=1}^{n} \frac{g(\mathbf{z}_{\theta(i)} \mid \mathbf{x}_i, \mathbf{q})}{c(\mathbf{z}_j \mid \mathbf{q})}. \tag{2.21}$$

No clutter but missed detections possible In this case all measurements are due to true objects but there is the possibility of missed detections, so $m \leq n$ and all valid data associations have the property that $\theta(i) = \theta(j) > 0 \Rightarrow i = j$. The probability of receiving no measurements is then

$$p(\varnothing \mid X, \mathbf{q}) = \prod_{i=1}^{n} \big(1 - p_d(\mathbf{x}_i \mid \mathbf{q})\big). \tag{2.22}$$

If $Z \neq \varnothing$ then the likelihood is

$$p(Z \mid X, \mathbf{q}) = \sum_{\theta} \prod_{i \mid \theta(i) > 0} p_d(\mathbf{x}_i \mid \mathbf{q}) g(\mathbf{z}_{\theta(i)} \mid \mathbf{x}_i, \mathbf{q}) \prod_{i \mid \theta(i) = 0} \big(1 - p_d(\mathbf{x}_i \mid \mathbf{q})\big) \tag{2.23}$$

$$= p(\varnothing \mid X, \mathbf{q}) \sum_{\theta} \prod_{i \mid \theta(i) > 0} \frac{p_d(\mathbf{x}_i \mid \mathbf{q}) g(\mathbf{z}_{\theta(i)} \mid \mathbf{x}_i, \mathbf{q})}{1 - p_d(\mathbf{x}_i \mid \mathbf{q})}. \tag{2.24}$$

Missed detections and clutter possible This is the most general case and captures all possible behavior. Data associations are functions from $\{1, \ldots, n\} \rightarrow \{0, \ldots, m\}$ with the property that $\theta(i) = \theta(j) > 0 \Rightarrow i = j$. The cases where $n = 0$ or $m = 0$ are handled above.

When $n > 0$ and $m > 0$ then the measurement likelihood is

$$p(Z \mid X, \mathbf{q}) = \sum_{\theta} \prod_{i|\theta(i)>0} p_d(\mathbf{x}_i \mid \mathbf{q}) g(\mathbf{z}_{\theta(i)} \mid \mathbf{x}_i, \mathbf{q}) \prod_{i|\theta(i)=0} \big(1 - p_d(\mathbf{x}_i \mid \mathbf{q})\big) \prod_{\substack{j|\theta(i)\neq j \\ \forall i \in \{1,\ldots,n\}}} c(\mathbf{z}_j \mid \mathbf{q}) \tag{2.25}$$

$$= p(Z \mid \varnothing, \mathbf{q})\, p(\varnothing \mid X, \mathbf{q}) \sum_{\theta} \prod_{i|\theta(i)>0} \frac{p_d(\mathbf{x}_i \mid \mathbf{q}) g(\mathbf{z}_{\theta(i)} \mid \mathbf{x}_i, \mathbf{q})}{\big(1 - p_d(\mathbf{x}_i \mid \mathbf{q})\big)\, c(\mathbf{z}_{\theta(i)} \mid \mathbf{q})}. \tag{2.26}$$

As is evident from the measurement likelihood functions above, the data association problem is computationally intractable for large problems. Returning to the multi-target methods presented in Section 2.1, the MHT tracks each possible association history separately. The approach from Dissanayake et al. [31] uses a number of heuristics to approximate the maximum likelihood association. JPDA performs this marginalization process over each object individually rather than over the full set. This assumes that the associations for each object are independent and removes the requirement that a detection cannot be generated from more than one object. The approach described above is the most general method and is, in some sense, the most technically correct because it explicitly considers a distribution over all valid data associations.

2.2.2 Key Mathematical Concepts

While a set-based representation offers several advantages over a vector-based approach, it requires some mathematics that are unfamiliar to most roboticists. In order to perform statistical inference over sets, we must define appropriate random variables and be able to perform operations, such as taking expectations, over these random variables.

Random Finite Sets

The main concept in FISST is that of a random finite set.

Definition 1 (Random finite set)**.** A random variable with realizations as finite sets. It is characterized by a discrete distribution over the number of elements in the set and a family of joint distributions that characterize the distribution of the elements, which are

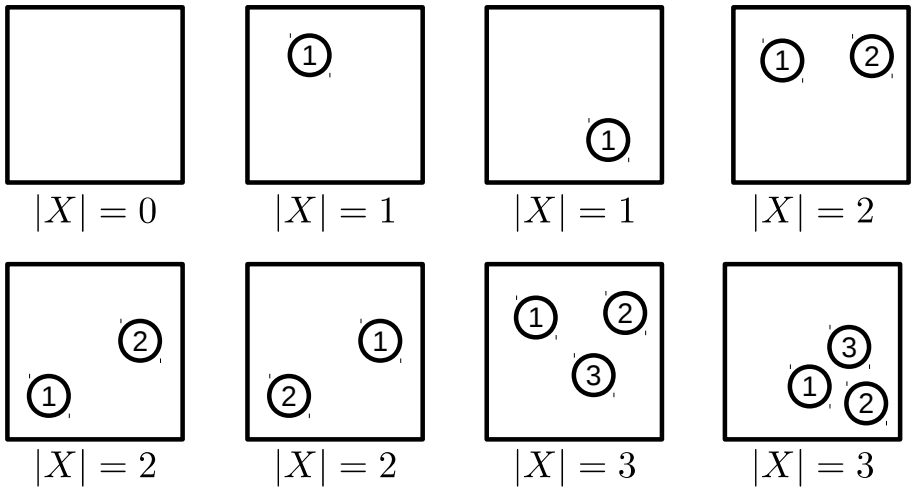


Figure 2: Examples of random finite sets with 0 to 3 elements drawn from the square environment. The two sets in the lower left are identical, as sets are equivalent under permutations of their elements, *i.e.*, $X = \{1, 2\} = \{2, 1\}$.

conditioned on the cardinality,

$$p(X) = p(|X| = n)\, p(X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \mid |X| = n). \tag{2.27}$$

In a robotic mapping setting, the map landmarks and the measurements are both represented as Random Finite Sets (RFSs). See Figure 2 for example realizations of RFSs. The goal is then to perform probabilistic inference over the map RFSs, using the evidence collected in the measurement RFSs. This differs from working with random vectors in several key ways: realizations of an RFS may have different cardinalities, so they cannot be added as a random vector would; sets are equivalent under permutations of the elements while random vectors are not; and the expected value of an RFS is not a set, but a density function.

Set Integral

To take into account the particular structure of RFSs, notions such as integration must be carefully handled. To that effect, Mahler [70] defines the set integral.

Definition 2 (Set Integral)**.** Let $f(X)$ be a real-valued function of sets. The set integral of

$f(X)$ is

$$\int f(X)\,\delta X = \sum_{n=0}^{\infty} \frac{1}{n!} \int f(\{\mathbf{x}_1, \ldots, \mathbf{x}_n\})\, d\mathbf{x}_1 \ldots d\mathbf{x}_n. \tag{2.28}$$

The set integral features a sum over the set cardinality, integrating over all possible sets for each cardinality. Note the $1/n!$ term, which accounts for the permutations of elements within the set X of size n, and the use of δ as the differential element.

Probability Distributions over Random Finite Sets

In particular, we are interested in functions $f(X)$ that represent probability distributions over RFSs. The derivation of a probability distribution over RFSs has its roots in point process theory. See Daley and Vere-Jones [21] or Stone et al. [95, Chapter 5.1] for an overview of the subject. As is the case with random vectors, it is not possible to maintain an arbitrary distribution over RFSs: we must make some assumptions to make the problem tractable.

The most natural assumption is that elements in an RFS are independently and identically distributed (i.i.d.). While this does disallow correlations between landmark locations, in general such correlations would be unknown. Even if there is some correlation between landmark or target locations, it is better to assume independence than to assume an incorrect correlation between objects when making probabilistic inferences. The likelihood of an RFS X with i.i.d. elements is

$$p(X) = |X|!\, p(|X|) \prod_{\mathbf{x} \in X} p(\mathbf{x}), \tag{2.29}$$

where $|\cdot|$ is the cardinality operator, the leading $|X|!$ is the number of permutations of elements in the set, $p(|X|)$ is the cardinality distribution, and $p(\mathbf{x})$ the probability of a landmark having state $\mathbf{x}$. For (2.29) to be a valid probability distribution, the set integral must be unity. Additionally, the nth term in (2.28) is the probability of there being n landmarks.

Probability Hypothesis Density

Even with this machinery, concepts such as the expected value of an RFS are not obvious. In a vector-based approach, the expected value is simply the weighted sum (or integral),

$$\mathbf{E}[\mathbf{x}] = \int_X p(\mathbf{x})\,\mathbf{x}\,d\mathbf{x}. \tag{2.30}$$

This operation is no longer well-defined in the case of a set, where there is no notion of addition for two sets.

Evaluating the mean of an RFS requires some results from point process theory [21, Chap. 5]. In particular, the kth order statistical moment of an RFS X, $m_{X,k}$, is:

$$m_{X,k}(\mathbf{x}_1, \ldots, \mathbf{x}_k) = \int p(\{\mathbf{x}_1, \ldots, \mathbf{x}_k\} \cup W)\,\delta W. \tag{2.31}$$

The first moment is the simplest and is equal to the mean of the RFS.

This has a more intuitive interpretation, called the probability hypothesis density (PHD). Let $\delta_X(\mathbf{x}) = \sum_{\mathbf{y}\in X} \delta_{\mathbf{y}}(\mathbf{x})$, where $\delta_{\mathbf{y}}(\mathbf{x})$ is the Kronecker delta function, and let $\mathbf{1}_S(\mathbf{x})$ is the indicator function. Define the first moment to be $v(\mathbf{x}) = m_{X,1}(\mathbf{x})$. Consider the integral of the first moment over a region S, as is done by Mahler [67, Theorem 2],

$$\begin{aligned}
\int_S v(\mathbf{x})\,d\mathbf{x} &= \int \mathbf{1}_S(\mathbf{x})\,v(\mathbf{x})\,d\mathbf{x} \\
&= \int \mathbf{1}_S(\mathbf{x}) \int p(\{\mathbf{x}\} \cup W)\,\delta W d\mathbf{x} \\
&= \iint \mathbf{1}_S(\mathbf{x})\,\delta_X(\mathbf{x})\,d\mathbf{x}\,p(X)\;\delta X \\
&= \int |X \cap S|\,p(X)\,\delta X.
\end{aligned} \tag{2.32}$$

This states that the integral of the PHD over a region S is equal to the expected number of landmarks within that region. In other words, the PHD is a density function over the state space of a landmark that describes the expected spatial density of landmarks.

Note that the PHD is *not* a probability density function. However, in the case of an

i.i.d. RFS, the PHD is a scalar multiple of the likelihood of landmark locations. The total expected number of landmarks is given by the integral of the PHD over the entire state space. Mahler [67, Theorem 4] also shows that a Poisson approximation to a general distribution over RFSs is optimal with respect to the Kullback-Liebler divergence when the intensity function is the PHD.

2.2.3 Estimation Using Random Finite Sets

With the mathematical tools outlined above, it is possible to perform online estimation using several different approaches. Each of the approaches below represents the distribution over RFSs in a different manner, with associated advantages and disadvantages. In an object detection and localization setting, RFSs naturally apply when the number of objects is unknown. Additionally, RFSs may be used to model the sensor measurements, as the number of measurements in a single scan varies over time due to target and sensor motion, false negative detections, and false positive measurements.

General Bayesian Filter

The most general formulation of the set-based estimation problem maintains a distribution over RFSs themselves [70]. Let x be the state of a single target, $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a set of n target states, and let $\mathcal{X}$ be the RFS of target states. Similarly, let z be a single measurement, $Z = \{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ be a set of m measurements, and $\mathcal{Z}$ be the RFS of measurements. Throughout this work, we will use lower case letters to indicate scalars and vectors, capital letters to indicate sets, and script letters to indicate random variables.

The goal of Bayesian inference is to maintain a distribution over potential target sets $\mathcal{X}$, using the collected measurements Z, sensor models, and target models to inform the updates. Let $f(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ be the single-target motion model and $f(X_t \mid X_{t-1})$ be the target set motion model, where the latter is more general as it allows the motion of targets to be correlated. Let $g(\mathbf{z} \mid \mathbf{x})$ be the single-target measurement model and $g(Z \mid X)$ be the multiple-target measurement model. Note that the single-target model assumes that a detection has been made while the set-based model is more general, allowing for missed

detections, false alarms, and unknown data association.

The most general formulation is the Bayesian filter, which, like the single-target case in (2.1), is based off of Bayes' rule

$$p(X|Z) = \frac{p(Z \mid X)p(X)}{p(Z)}. \tag{2.33}$$

Let X denote the RFS of target states and Z the RFS of measurements. In estimation problems there is not typically an expression for the measurement likelihood $p(Z)$. Instead, we have the conditional likelihood of the measurements given the target states, $p(Z \mid X)$. The denominator in (2.33) may be rewritten as a marginal distribution,

$$p(Z) = \int p(X, Z)\, \delta X = \int p(Z \mid X)p(X)\, \delta X.$$

Combining these results gives us the expression for the Bayesian filter,

$$p_{t|t-1}(X \mid Z_{0:t-1}) = \int f(X \mid \Xi)p(\Xi \mid Z_{0:t-1})\, \delta\Xi \tag{2.34}$$

$$p_t(X \mid Z_{0:t}) = \frac{g(Z \mid X)p(X \mid Z_{0:t-1})}{\int g(Z \mid X)p(X \mid Z_{0:t-1})\, \delta X} \tag{2.35}$$

While in general it is not possible to maintain the full distribution over RFSs, it is possible to approximate it with a set of weighted particles, with each particle having an associated set of landmarks [102, Sec. II.E]. When particles are propagated forward in time, each landmark has a probability of being removed from the set, and there is a probability of adding additional landmarks to the set.

PHD Filter

The most basic approach to estimation using RFSs is the PHD filter. This filter recursively updates the mean of the distribution over RFSs, making it the analog of the mean update term in the Kalman filter. In the case of the Kalman filter all distributions are Gaussian, while in this case they are all Poisson RFSs.

Definition 3 (Poisson RFS)**.** An RFS is said to be Poisson if the elements are i.i.d. and the cardinality follows a Poisson distribution. The likelihood of such an RFS is

$$p(X) = e^{-\lambda} \prod_{\mathbf{x} \in X} v(\mathbf{x}), \tag{2.36}$$

where $\lambda = \int v(\mathbf{x})\, d\mathbf{x}$.

The PHD filter was first derived by Mahler [67]. In its most generic form, it allows for arbitrary target motion, including the spawning (birth) of new targets and the disappearance of existing targets. In order to derive the PHD filter equations, Mahler [67] made the following assumptions:

A1. targets move and generate measurements independently;

A2. birth and surviving RFSs are independent;

A3. the clutter RFS is Poisson and independent of true measurements;

A4. prior and predicted multitarget RFSs are Poisson.

Let $f(\mathbf{x} \mid \xi)$ be the likelihood of a single target moving from state ξ to state $\mathbf{x}$. Let $B(\xi)$ be a Poisson RFS of targets spawned by existing targets and let $b(\mathbf{x} \mid \xi)$ be its PHD. Let B be a Poisson RFS of new targets that enter the environment and let $b(\mathbf{x})$ be its PHD. Let $p_s(\mathbf{x})$ be the likelihood of a target with state $\mathbf{x}$ surviving from one time step to the next. As a matter of notation, we define the inner product between two real-valued functions $\langle a, b \rangle$ to be

$$\langle a, b \rangle = \int a(\mathbf{x}) b(\mathbf{x})\, d\mathbf{x},$$

or $\langle a, b \rangle = \sum_{k=0}^{\infty} a(k) b(k)$ for real-valued sequences.

Then the PHD prediction equation is

$$v_{t|t-1}(\mathbf{x}) = b_{t|t-1}(\mathbf{x}) + \int \big(b_{t|t-1}(\mathbf{x} \mid \xi) + p_s(\xi) f(\mathbf{x} \mid \xi)\big) v_{t-1}(\xi)\, d\xi. \tag{2.37}$$

Let $p_d(\mathbf{x} \mid \mathbf{q})$ be the likelihood of a sensor with state $\mathbf{q}$ detecting a target with state $\mathbf{x}$. Let $g(\mathbf{z} \mid \mathbf{x}, \mathbf{q})$ be the likelihood of sensor with state $\mathbf{q}$ receiving a measurement $\mathbf{z}$ from a target with state $\mathbf{x}$ given that a detection is made. Let $C(\mathbf{q})$ be the Poisson RFS of clutter measurements and $c(\mathbf{z} \mid \mathbf{q})$ be its PHD. Then the PHD corrector equation is

$$v_t(\mathbf{x}) = \big(1 - p_d(\mathbf{x} \mid \mathbf{q})\big) v_{t|t-1}(\mathbf{x}) + \sum_{\mathbf{z} \in Z_t} \frac{\psi_{\mathbf{z},\mathbf{q}}(\mathbf{x}) v_{t|t-1}(\mathbf{x})}{c(\mathbf{z} \mid \mathbf{q}) + \langle \psi_{\mathbf{z},\mathbf{q}},\, v_{t|t-1} \rangle} \tag{2.38}$$

$$\psi_{\mathbf{z},\mathbf{q}}(\mathbf{x}) = g(\mathbf{z} \mid \mathbf{x}, \mathbf{q})\, p_d(\mathbf{x} \mid \mathbf{q}). \tag{2.39}$$

Here $\psi_{\mathbf{z},\mathbf{q}}(\mathbf{x})$ is the likelihood of sensor with state $\mathbf{q}$ receiving a measurement $\mathbf{z}$ from a target with state $\mathbf{x}$.

Gaussian Mixture PHD Filter As is the case with single target estimation strategies, it is not possible to maintain a generic density function over the state space of the targets. One approach to get around this limitation, from Vo and Ma [101], is known as the Gaussian Mixture PHD (GM-PHD) filter and represents the PHD as a weighted mixture of Gaussians. In this, they assume that the target motion model and sensor model are linear Gaussian, that the survival and detection probabilities are state independent or are weighted mixtures of Gaussians, and that all PHDs are weighted mixtures of Gaussians.

The net result is that the GM-PHD filter becomes a sequence of Kalman filter updates. In the update step, each component in the prior generates a new component in the predicted PHD for each component in the survival probability and target spawning PHD. Additionally, the components in the birth PHD are added to these other components. So if there are $J_{t-1|t-1}$ components in the prior, S components in the survival probability, P components in the spawning PHD, and B components in the birth PHD, then there are $J_{t|t-1} = B + (S + P)J_{t-1|t-1}$ components in the predicted PHD. Each of these individual components evolves according to the update rules for the Kalman filter, and thus can be swapped out in favor of the EKF or UKF if there is a non-linear target motion model.

Similarly, the update equation is a sequence of Kalman updates on the individual components of the GM. Each component in the predicted PHD generates a new component

for each component in the detection likelihood and for each measurement. So if there are D components in the detection likelihood function and Z measurements, there will be $J_{t|t} = (D + Z)J_{t|t-1}$ components in the posterior PHD.

As is evident, the number of components can grow rapidly over time. To keep the computation burden bounded, the number of components in the mixture model must be bounded. This can be achieved by pruning components with very low weights and by merging components that are sufficiently close to one another. See Vo and Ma [101, Table II] for a simple pruning and merging strategy.

Sequential Monte Carlo PHD Filter Another common approach is to represent the PHD as a set of weighted particles. This approach from Vo et al. [102] is known as the Sequential Monte Carlo PHD (SMC-PHD) filter. This is essentially equivalent to a standard particle filter, except that the weights of the particles are not normalized to have unit weight. The SMC-PHD filter offers one key advantage over the GM-PHD filter: it allows for arbitrary target and sensor likelihood functions. In particular, this is useful in instances where the probability of detection is non-zero only within a finite footprint for detection likelihoods.

CPHD Filter

While the PHD filter is attractive due to its low computational complexity and relatively simple implementation, it suffers from two potential drawbacks. Firstly, as pointed out by Erdinc et al. [33], the PHD filter deals poorly with false negatives, drastically decreasing the likelihood of a target being within a given region if no detection is made. Secondly, the target cardinality estimate has high variance, particularly when tracking a large number of targets, due to the fact that the mean and variance of a Poisson distribution are equal. To get around these issues, Mahler [69] developed the CPHD filter.

The CPHD filter makes the same assumptions as the PHD filter, except instead of Poisson RFSs, everything is assumed to be an i.i.d. cluster process. This allows for an arbitrary discrete distribution over target cardinality and improved performance with missed detections, at the cost of an increase in computational complexity. Two drawback are that the maximum number of targets that can be tracked is fixed a priori and the cardinality

estimate will be biased if the true number of targets is close to the maximum value, as noted by Vo et al. [103].

Before stating the CPHD prediction and update rules, we must define a few variables. Let $p_t(n)$ be the likelihood of n targets at time t, $p_{\gamma,t}(n)$ be the cardinality distribution of the birth process Γ at time t, and $p_{K,t}(n)$ be the cardinality distribution of the clutter process. Let $\binom{\ell}{j}$ be the binomial coefficient $(\ell!/(\ell-j)!j!)$ and P_j^ℓ be the permutation coefficient $(\ell!/(\ell-j)!)$.

The CPHD filter prediction equations, following the derivation of Vo et al. [103], are

$$p_{t|t-1}(n) = \sum_{j=0}^{n} p_\Gamma(n-j)\Pi_{t|t-1}[v_{t-1}, p_{t-1}](j) \tag{2.40}$$

$$v_{t|t-1}(\mathbf{x}) = \int p_S(\xi) f(\mathbf{x} \mid \xi) v_{t-1}(\xi)\, d\xi + \gamma(\mathbf{x}) \tag{2.41}$$

where

$$\Pi_{t|t-1}[v,p](j) = \sum_{\ell=j}^{\infty} \binom{\ell}{j} \frac{\langle p_S,\, v\rangle^j \langle 1-p_S,\, v\rangle^{\ell-j}}{\langle 1,\, v\rangle^\ell} p(\ell)$$

is the probability of j targets surviving from time $t-1$ to t.

The update equations are more complicated as the cardinality and PHD updates are coupled,

$$p_t(n) = \frac{\Upsilon^0[v_{t|t-1}, Z_t](n) p_{t|t-1}(n)}{\left\langle \Upsilon^0[v_{t|t-1}, Z_t],\, p_{t|t-1}\right\rangle} \tag{2.42}$$

$$\begin{aligned} v_t(\mathbf{x}) = {} & \frac{\left\langle \Upsilon^1[v_{t|t-1}, Z_t],\, p_{t|t-1}\right\rangle}{\left\langle \Upsilon^0[v_{t|t-1}, Z_t],\, p_{t|t-1}\right\rangle}(1-p_d(\mathbf{x}))v_{t|t-1}(\mathbf{x}) \\ & + \sum_{z\in Z_t} \frac{\left\langle \Upsilon^1[v_{t|t-1}, Z_t \setminus \{z\}],\, p_{t|t-1}\right\rangle}{\left\langle \Upsilon^0[v_{t|t-1}, Z_k],\, p_{t|t-1}\right\rangle} \psi_{\mathbf{z},\mathbf{q}}(\mathbf{x}) v_{t|t-1}(\mathbf{x}), \end{aligned} \tag{2.43}$$

where

$$\Upsilon^u[v,Z](n) = \sum_{j=0}^{\min(|Z|, n-u)} (|Z|-j)!\, p_K(|Z|-j) P_{j+u}^n \frac{\langle 1-p_d,\, v\rangle^{n-(j+u)}}{\langle 1,\, v\rangle^n} e_j(\Xi(v,Z)) \tag{2.44}$$

$$\Xi(v,Z) = \{\langle v,\, \psi_{\mathbf{z},\mathbf{q}}\rangle \mid \mathbf{z} \in Z\}. \tag{2.45}$$

Here $\Upsilon^u[v, Z](n)$ is proportional to the likelihood of target set Z given that there are n targets and u targets are not detected. The function $e_j(\Xi)$ is the elementary symmetric polynomial of order j,

$$e_j(\Xi) = \sum_{\substack{S \subseteq \Xi \\ |S|=j}} \prod_{\xi \in S} \xi, \tag{2.46}$$

which can be computed efficiently using Vieta's formula, as noted by Vo et al. [103], yielding a total complexity for the CPHD filter of $\mathcal{O}(|Z|^2 \log |Z|)$ as opposed to the $\mathcal{O}(|Z|)$ updates for the PHD filter. When the number of targets is very large, the CPHD filter will be significantly slower.

2.2.4 Literature Review

Multi-target tracking has also been addressed extensively in the radar tracking community; Pulford [82] provides a taxonomy of techniques. Recently the use of random finite sets has been adopted in mobile robotics, being used for feature-based mapping by Mullane et al. [76, 77]. Lundquist et al. [65] use a PHD filter for extended objects (*i.e.*, objects that return multiple measurements) to create an obstacle map for a vehicle. Atanasov et al. [4] present an approach to localize a robot in a semantic map using an approximation algorithm to solve the data association (2.26). Other applications of FISST in robotic mapping, target tracking, and SLAM are presented in [2].

2.3 Active Information Gathering

Information-based control is a common tool for information gathering tasks. The intuition is to drive the team of robots in a way that minimizes some measure of uncertainty about the environment state. This section provides a brief summary of uncertainty measures and a survey of the current literature on information-based control in a variety of settings.

2.3.1 Uncertainty Measures

There are many ways to quantify the uncertainty associated with a generic random variable, including RFSs or distributions approximated by a histogram or particle filter. With parameterized distributions, such as the Gaussian, there are specialized tools that take advantage

of the functional form of the distribution.

Gaussian Distribution

With a Gaussian distribution, the covariance matrix fully characterizes the spread of the distribution. From the theory of optimal experiment design [9, 81], there are several standard optimality criteria that map a covariance matrix to a scalar while retaining useful statistical properties. The three most widely used criteria are:

- A-optimality minimizes the average variance,

$$\frac{1}{n}\text{trace}(\Sigma) = \frac{1}{n}\sum_{k=1}^{n}\lambda_k \tag{2.47}$$

  where n is the dimension of the covariance matrix Σ and λ_k is its kth eigenvalue.

- D-optimality minimizes the volume of the covariance ellipsoid,

$$\det(\Sigma)^{1/n} = \exp\left(\frac{1}{n}\sum_{k=1}^{n}\log(\lambda_k)\right). \tag{2.48}$$

- E-optimality minimizes the maximum eigenvalue of the covariance matrix, Σ,

$$\max_k(\lambda_k). \tag{2.49}$$

General Distributions

Uncertainty measures for general distributions come from information theory. The most common measures are due to Shannon [91]. Cover and Thomas [20] provide an excellent summary of information theory and provide many useful identities and inequalities.

The entropy of a continuous random variable $\mathcal{X}$ is

$$H[\mathcal{X}] = -\int p(\mathbf{x})\log p(\mathbf{x})\,d\mathbf{x}, \tag{2.50}$$

where the integral is replaced by a sum for a discrete random variable.

The conditional entropy of a random variable $\mathcal{X}$ given another variable $\mathcal{Z}$ is

$$H[\mathcal{X} \mid \mathcal{Z}] = -\int p(\mathbf{z}) \int p(\mathbf{x} \mid \mathbf{z}) \log p(\mathbf{x} \mid \mathbf{z})\, d\mathbf{x} d\mathbf{z} = -\iint p(\mathbf{x}, \mathbf{z}) \log p(\mathbf{x} \mid \mathbf{z})\, d\mathbf{x} d\mathbf{z}. \quad (2.51)$$

The difference between these values yields the mutual information between $\mathcal{X}$ and $\mathcal{Z}$, which is a way to quantify the amount of dependence between two random variable. Mutual information is computed as

$$\begin{aligned} I[\mathcal{X}; \mathcal{Z}] &= H[\mathcal{X}] - H[\mathcal{X} \mid \mathcal{Z}] && (2.52)\\ &= H[\mathcal{Z}] - H[\mathcal{Z} \mid \mathcal{X}] && (2.53)\\ &= \iint p(\mathbf{x}, \mathbf{z}) \log \frac{p(\mathbf{x}, \mathbf{z})}{p(\mathbf{x}) p(\mathbf{z})}\, d\mathbf{x} d\mathbf{z}. && (2.54) \end{aligned}$$

Note that if $\mathcal{X}$ and $\mathcal{Z}$ are independent, then the term inside the log in (2.54) will be unity so the integral will be zero.

Entropy of a Poisson RFS The entropy of a Poisson RFS X follows from substituting the likelihood function (2.36) into the standard Shannon definition of entropy, replacing the integral with a set integral. Recalling that $\lambda = \int v(\mathbf{x})\, d\mathbf{x}$, we see that

$$\begin{aligned} H[\mathcal{X}] &= -\int p(X) \log p(X)\, \delta X \\ &= -e^{-\lambda} \sum_{n=0}^{\infty} \frac{1}{n!} \int \prod_{i=1}^{n} v(\mathbf{x}_i) \left[-\lambda + \sum_{j=1}^{n} \log v(\mathbf{x}_j) \right] d\mathbf{x}_1 \ldots d\mathbf{x}_n \\ &= -e^{-\lambda} \sum_{n=0}^{\infty} \frac{1}{n!} \left[-\lambda \left(\int v(\mathbf{x})\, d\mathbf{x} \right)^n + n \left(\int v(\mathbf{x})\, dx \right)^{n-1} \left(\int v(\mathbf{x}) \log v(\mathbf{x})\, d\mathbf{x} \right) \right] \\ &= \left(\lambda - \int v(\mathbf{x}) \log v(\mathbf{x})\, d\mathbf{x} \right) \sum_{n=0}^{\infty} \frac{1}{n!} \lambda^n e^{-\lambda} \\ &= \lambda - \int v(\mathbf{x}) \log v(\mathbf{x})\, d\mathbf{x}. \quad (2.55) \end{aligned}$$

This may also be written using the normalized density, $\bar{v}(\mathbf{x}) = \lambda^{-1} v(\mathbf{x})$, as,

$$\begin{aligned} H[\mathcal{X}] &= \lambda - \lambda \int \bar{v}(\mathbf{x}) [\log \lambda + \log \bar{v}(\mathbf{x})] \, dx \\ &= \lambda - \lambda \log \lambda - \lambda \int \bar{v}(\mathbf{x}) \log \bar{v}(\mathbf{x}) \, dx \\ &= \lambda - \lambda \log \lambda + \lambda H[\bar{v}(\mathbf{x})], \end{aligned} \tag{2.56}$$

where $H[\bar{v}(\mathbf{x})]$ is the Shannon entropy of the probability density function $\bar{v}(\mathbf{x})$.

2.3.2 Information-Based Control

Information-based control has seen a lot of attention in recent years as a way of driving robots to localize and track targets. Mutual information is a common objective to use in target tracking problems. Hoffmann and Tomlin [45] and Julian et al. [51] use mutual information to localize a stationary target and explore unknown environments using a team of robots, assuming limited dependence between robots to achieve scalability. Hollinger et al. [47] use an information-based objective function to perform autonomous ship inspection with an AUV platform. The robot may also move to maximize the immediate information gain, a strategy sometimes known as "information surfing" [19]. Julian et al. [50] use the gradient of mutual information to drive multiple robots for state estimation tasks, a strategy sometimes known as "information surfing" [19]. Julian et al. [52] and Souza et al. [93] utilize mutual information to drive a single robot to explore an unknown environment in order to build a map. Charrow et al. [11, 12] use mutual information to drive a team of robots equipped with range-only sensors to track a single moving target in real time and to detect and localize an unknown number of targets with known data association. All of these approaches assume that the data association is known and all but Charrow et al. [12] assume that the number of targets is known.

Our control policy for active perception for multi-target tracking builds on the literature on receding horizon control and model predictive control. Mayne and Michalska [72] provide a survey of receding horizon control and Mayne et al. [73] provide a survey of model predictive control, including applications in a variety of domains. Jadbabaie [48] utilizes model

predictive control to follow trajectories with UAVs. The work of Ryan [89] is particularly relevant as it uses model predictive control in an information gathering setting, using a small team of UAVs to localize and track a moving target. We adapt this work to the multi-target, active estimation problem to consider actions over an extended time horizon, rather than a simple myopic exploration strategy.

One common approach to robot control for active estimation is to maximize mutual information between the target locations and the robots' measurements. Grocholsky [41], Bourgault et al. [6], and Cole [18] consider information-theoretic control of robot teams for exploration and tracking tasks using the Decentralized Data Fusion (DDF) architecture to handle inter-agent communication. In particular, Cole [18] examines the scenario where the number of targets is unknown, deriving equations similar to those of the PHD filter but using a very conservative data fusion approach. Stranders et al. [96] and Delle Fave et al. [30] use the max-sum algorithm for decentralized control computations and DDF to share beliefs about target locations. However, all of these approaches make restrictive assumptions on the form of the distribution over targets, often requiring Gaussian distributions. None of these approaches can handle the case of an unknown number of targets.

There is a relatively limited body of work on active control for target localization based on the RFS framework, with the exception of work by Ristic and Vo [85] and Ristic et al. [87] to maximize information using Rényi's definition. Ristic and Vo [85] track the target positions using samples from the distribution over RFSs, as in Section 2.2.3. In this work, the measurement model involves a summation over all possible data associations and the authors present simulation results of a single robot seeking three targets in an open environment. Ristic et al. [87] use the SMC-PHD filter from Section 2.2.3 to track target positions. This is most similar to the framework presented in this dissertation, but the authors only present work using a single robot selecting from eight motion primitive to track five objects in simulation.

Similarly, the problem of placing static sensors, rather than controlling dynamic ones, has been treated in a number of works. Notably, the property of submodularity of mutual

Table 2: Table of symbols.

$\cdot^r$	Robot index	R	Number of robots
$\mathbf{q}$	Robot pose	Q	Action set
$v(\mathbf{x})$	Target PHD	λ	Expected # targets
$\mathbf{x}$	Target pose	$\mathbf{z}$	Measurement
X	Target set	Z	Measurement set
$\mathcal{X}$	Target random variable	$\mathcal{Z}$	Measurement random variable
$p_d(\mathbf{x} \mid \mathbf{q})$	Probability of detection	$g(\mathbf{z} \mid \mathbf{x}, \mathbf{q})$	Measurement likelihood
$c(\mathbf{z} \mid \mathbf{q})$	Clutter PHD	μ	Expected clutter rate
$\cdot_t$	Time index	T	Time horizon
ϵ	Termination criterion	L	Number of length scales

information was used by Krause and Guestrin [56], Krause et al. [58] to prove near-optimal static placement. The technique was extended to Gaussian processes by Krause et al. [59]. Different approximations were derived for the static sensor placement problem by Choi and How [14, 15], Choi et al. [16], and an informative trajectory planning algorithm was presented by Choi and How [13]. Unfortunately, our algorithm cannot make use of these near-optimality results because the sequential updating of our distribution destroys the submodularity property of mutual information. Other works concentrate on specific models of target positions or environmental fields. For example the algorithm by Lynch et al. [66] drives robots to decrease the error variance of a distributed Kalman filter estimate of a Gaussian environmental field. By contrast, our algorithm does not make assumptions about the Gaussianity of the distribution of targets or hazards.

Chapter 3

Active Detection and Localization of a Small Number of Targets

Teams of mobile robots may be used in many applications to gather information about unknown, hazardous environments, taking measurements at multiple locations while keeping humans out of harm's way. It would be useful, for example, to deploy a team of robots to search for survivors in a building after an earthquake or other disaster, where the number of survivors is unknown a priori. In this scenario the building may be structurally unstable and there may be fires or exposed live electrical wires in the environment, all of which may cause harm to rescuers and robots. As multiple robots will likely fail, it is advantageous to use low-cost platforms. However, such platforms have limited capabilities, and thus the control strategy should make minimal assumptions about the sensors and environment.

This chapter proposes an approach to tackle this problem that employs a coarse, high-level sensor model, wherein sensors only provide binary information indicating whether they have detected a target or not and hazards are only "detected" through robot failures. With such coarse sensing capabilities it is natural to also use a coarse representation of the environment, decomposing the space into a collection of cells. The goal is then to determine which cells contain objects (*e.g.*, trapped survivors) or hazards (*e.g.*, fires) and which cells are empty. Bayesian estimators maintain distributions of the object and hazard locations

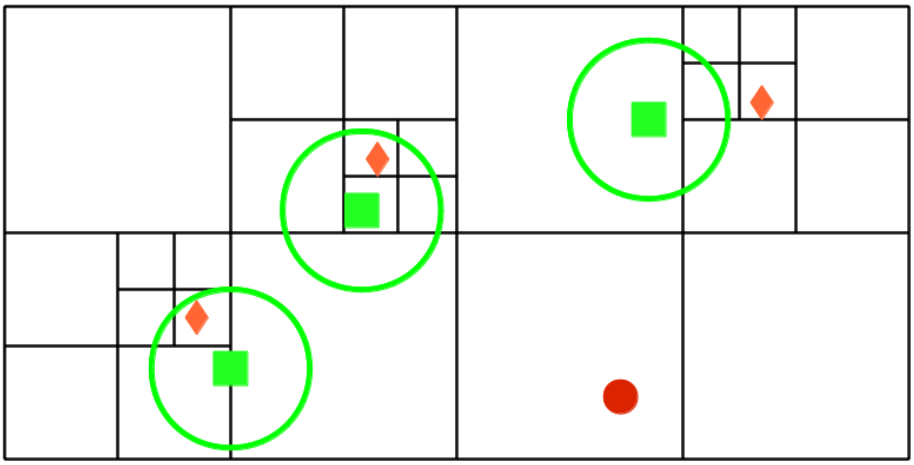


Figure 3: Illustration of our multi-robot multi-target localization algorithm. The robots (green squares) estimate the locations of targets (orange diamonds) and hazards (red dots) with high resolution by adaptively refining a cellular decomposition of the environment, despite having noisy sensors. The robots move to improve their estimate of the target locations while avoiding the estimated hazard locations by following the gradient of mutual information. The robots' finite sensor footprints (green circles) allow for decentralized estimation and control computations.

using the detections and failures of the robots in the team. The resolution of these cells is dynamically updated to provide finer localization of targets with limited computational resources. Using these estimates, the decentralized control algorithm moves the team of robots in the direction of greatest immediate information gain, a strategy sometimes called "information surfing" [41]. More precisely, the controller moves the robots along the gradient of mutual information of target locations and measurements with respect to the positions of the robots. This implicitly tends to drive the robots to avoid hazardous areas as a failed robot provides no information, naturally merging the objectives of localizing targets and avoiding hazards.

The research in this chapter was originally published in [26, 27, 90].

3.1 Problem Formulation

Consider a situation where n robots move in a bounded, planar environment $E \subset \mathbb{R}^2$. Robot i is at position $\mathbf{q}_t^i \in E$ at time t, and the positions of all the robots can be written as the stacked vector $\mathbf{q}_t = [(\mathbf{q}_t^1)^T \dots (\mathbf{q}_t^n)^T]^T$. Each robot is equipped with a binary sensor which gives measurements $z^i \in \{0, 1\}$ indicating whether or not the sensor has detected a target.

Robots can also detect the failure status of other robots, $f^i \in \{0,1\}$, where $f^i = 1$ indicates that robot i has failed. Let the vector of sensor measurements be given by $\mathbf{z} = [z^1, \ldots, z^n]^T$, where $z \in \{0,1\}^n = \mathcal{Z}$, and the vector of all failure statuses by $\mathbf{f} = [f^1, \ldots, f^n]^T \in \{0,1\}^n$.

3.1.1 Map Representation

Finite set statistics (FISST) circumvents the issue of data association in target tracking by not implicitly (or explicitly) labeling individual targets. Rather than random vectors, FISST is based on *random finite sets* (RFSs), which are sets containing a random number of random elements describing the locations of each target. In this scenario, with the environment being represented by a collection of discrete cells, an RFS will be a set of labels of occupied cells. Due to the discretization of the environment in our case, the set integral will reduce to a finite sum. Also, the restriction that elements in an RFS be unique means that only one target may be within each cell, requiring the minimum cell size to be smaller than the minimum separation between objects. By employing an adaptive discretization of the environment, individual targets may be localized with high precision while empty areas are represented by a small number of large cells. Let the discretization representing target locations be denoted $\{E_j^s\}_{j=1}^{m_T} \subset E$, where m_T is the number of cells, and a set of target locations be $X \in \mathcal{X}$, where $\mathcal{X}$ is the RFS for a given discretization. Similarly, another discretization $\{E_j^h\}_{j=1}^{m_H} \subset E$ is used to represent the locations of hazards within the environment and a set of cell labels drawn from this discretization is denoted $H \in \mathcal{H}$.

3.1.2 Sensor Models

As previously mentioned, the robots have a chance of failure due to hazards in the environment. Let the probability of robot i, with pose $\mathbf{q}^i$, failing due to a hazard in cell E_j^h be modeled by $p(f^i = 1 \mid j \in H, \mathbf{q}^i) \approx \alpha(\mathbf{q}^i, E_j^h)$ while $p(f^i = 1 \mid j \notin H, \mathbf{q}^i) = 0$. We assume that robot failures are conditionally independent given the locations of the hazards so

$$p(f^i = 0 \mid H, \mathbf{q}^i) = (1 - p_{\mathrm{f}}) \prod_{j \in H} p(f^i = 0 \mid j, \mathbf{q}^i), \tag{3.1}$$

since the only way to not have a failure is to not fail due to any of the individual hazards or due to some other failure with probability $p_{\text{f}} (\ll 1)$. The probability of failure is then the additive complement of (3.1).

When a robot has failed, it provides no further information about the location of targets, leading to the conditional probability $p(z^i = 1 \mid f^i = 1, X, \mathbf{q}^i) = 0$. If a sensor is still functional, the detection equations are analogous to that of the hazards, beginning with $p(z^i = 1 \mid f^i = 0, j \in X, \mathbf{q}^i) \approx \mu(q^i, E^s_j)$ and $p(z^i = 1 \mid f^i = 0, j \notin X, \mathbf{q}^i) = 0$. The detections of each target are also conditionally independent given the target locations so

$$p(z^i = 0 \mid f^i = 0, X, H, \mathbf{q}^i) = (1 - p_{\text{fp}}) \prod_{j \in X} p(z^i = 0 \mid f^i = 0, j, \mathbf{q}^i), \tag{3.2}$$

where p_{fp} is the probability of a false positive reading.

The failure model, $p(f^i = 1 \mid j \in H, \mathbf{q}^i)$, and sensor model, $p(z^i = 1 \mid f^i = 0, j \in X, \mathbf{q}^i)$, of the robots have several key properties. First, real sensors have a finite field of view, so these models should have compact support. Let F^i be the set of labels of cells within the footprint of robot i and consider the subset of RFSs containing targets in F^i, $\mathcal{V}^i = \{X \in \mathcal{X} \mid x \in F^i \; \forall x \in X\}$. This is found using the projection $r^i_T : \mathcal{X} \to \mathcal{V}^i$ given by $r^i_T(X) = X \cap F^i$. Note that this map is surjective but not injective as long as F^i is a proper subset of E, so no inverse mapping exists. The right inverse still exists, where $r^i_T((r^i_T)^{-1}(V)) = V$ but $(r^i_T)^{-1}(r^i_T(X)) \neq X$. The right inverse of the projection is $(r^i_T)^{-1}(V) = \{X \mid r^i_T(X) = V\}$, which returns multiple values in general. Let W^i be the analogous neighborhood in the hazard grid with projection r^i_H.

Second, the features may be located anywhere within the cell. Given this, the probability of failure due to a hazard in cell E^h_j is given by

$$\alpha(\mathbf{q}^i, E^h_j) = \int_{E^h_j} g_h(\mathbf{q}^i, \mathbf{x}) p(\mathbf{x}) \, d\mathbf{x} \approx \frac{1}{m} \sum_{k=1}^{m} g_h(\mathbf{q}^i, \mathbf{e}^h_{j,k}), \tag{3.3}$$

where $g_h(\mathbf{q}^i, \mathbf{x})$ is a function describing the probability of failure due to a hazard at location $\mathbf{x}$ and $p(\mathbf{x})$ is a distribution of the location of the hazard in the cell. This integral is

approximated by a sum over a set of m points in the cell, $\{\mathbf{e}^h_{j,k}\}_{k=1}^m \in E^h_j$, which, given no available information beyond our binary failure readings, are distributed uniformly over cell. The simplest approach is to use the cell centroids. However, multiple points should be used for cells that are large compared to the sensor field of view.

Analogously the probability of detection is

$$\mu(\mathbf{q}^i, E^s_j) \approx \frac{1}{m} \sum_{k=1}^{m} g_s(\mathbf{q}^i, \mathbf{e}^s_{j,k}), \tag{3.4}$$

where $\{\mathbf{e}^s_{j,k}\}_k \in E^s_j$ is the set of points in cell E^s_j. This integration over the cell naturally takes into consideration the fraction of the cell that is visible to the sensor: if only a small portion is visible then μ will be low since most terms in the sum will be zero, while if the robot can see most of the cell then μ will be larger. However, the integration does not take into account the area viewed during previous time steps, as was noted by Waharte et al. [105].

Failures of multiple robots are assumed to be conditionally independent of one another given the positions of the hazards so that,

$$p(\mathbf{f} \mid X, H, Q) = \prod_i p(f^i \mid H, \mathbf{q}^i), \tag{3.5}$$

where $p(f^i \mid H, \mathbf{q}^i)$ comes from (3.1) and Q is the set of robot poses. Similarly, the robots' sensor measurements are conditionally independent given the locations of the targets, so that

$$p(\mathbf{z} \mid \mathbf{f}, X, H, Q) = \prod_i p(z^i \mid f^i, X, \mathbf{q}^i), \tag{3.6}$$

where $p(z^i \mid f^i, X, \mathbf{q}^i)$ comes from (3.2). Finally, marginalizing over the possible failure states yields the detection model

$$p(\mathbf{z} \mid X, H, Q) = \prod_i \sum_{f^i \in \{0,1\}} p(z^i \mid f^i, X, \mathbf{q}^i) p(f^i \mid H, \mathbf{q}^i). \tag{3.7}$$

3.1.3 Communication

A communication protocol is necessary in order to decentralize the exploration task. The simplest approach to the problem, employed here, uses the standard disk model, where a robot is able to communicate with all other robots within some disk around its current pose. Algorithm 1 outlines this approach, where robots exchange measurement and pose histories with all neighboring robots. The robots then use these measurements to update their estimate of the target and hazard locations. Note that it is not necessary to send the history of the failure status of the robots, as the robot would be unable to communicate if it had failed prior to the current time t. Section 3.2 provides a justification for the incorporation of old measurements.

Algorithm 1 Communication

```
for All robots, i do
    Discover robots in communication range, N^i
    for j ∈ N^i do
        Look up time of last communication, τ_j
        if τ_j < t then
            Send q^i_{τ_j:t}, z^i_{τ_j:t}, f^i_t
            Receive q^j_{τ_j:t}, z^j_{τ_j:t}, f^j_t
            τ^j ← t
        end if
    end for
    Update Bayesian filter using new measurements
end for
```

3.2 Bayesian Estimation

As the sensors explore the environment and exchange measurements, a recursive Bayesian filter makes use of the collected information in order to estimate the target and hazard locations. Let $\varphi_t(X) = p(X \mid \mathbf{z}_{1:t}, \mathbf{f}_{1:t}, Q_{1:t})$ be the estimated distribution over target sets at time t and $\psi_t(H) = p(H \mid \mathbf{z}_{1:t}, \mathbf{f}_{1:t}, Q_{1:t})$ be the estimate of the hazard set distribution. In this section, the dependence of the sensor and failure models on the poses of the robots will be omitted for brevity.

Theorem 1 (Bayesian Filtering). *The distributions for hazards and events, given all information up to time t, are independent with $p(X, H \mid \mathbf{z}_{1:t}, \mathbf{f}_{1:t}) = \varphi_t(X)\psi_t(H)$, assuming that h and s are not deterministically linked, and that their initial distributions are independent, $p(X, H) = \varphi_0(X)\psi_0(H)$. Furthermore, $\varphi_t(X)$ and $\psi_t(H)$ can be computed recursively with the Bayesian filters*

$$\varphi_t(X) = \frac{p(\mathbf{z}_t \mid \mathbf{f}_t, X)\varphi_{t-1}(X)}{\sum_{X \in \mathcal{X}} p(\mathbf{z}_t \mid \mathbf{f}_t, X)\varphi_{t-1}(X)}, \tag{3.8}$$

and

$$\psi_t(H) = \frac{p(\mathbf{f}_t \mid H)\psi_{t-1}(H)}{\sum_{H \in \mathcal{H}} p(\mathbf{f}_t \mid H)\psi_{t-1}(H)}. \tag{3.9}$$

In the case that the targets and the hazards are deterministically linked, the Bayesian filter update for the distribution is given by

$$p(X \mid \mathbf{z}_{1:t}, \mathbf{f}_{1:t}) = \frac{p(\mathbf{z}_t \mid \mathbf{f}_t, X)p(\mathbf{f}_t \mid X)p(X \mid \mathbf{z}_{1:t-1}, \mathbf{f}_{1:t-1})}{\sum_{X \in \mathcal{X}} p(\mathbf{z}_t \mid \mathbf{f}_t, X)p(\mathbf{f}_t \mid X)p(X \mid \mathbf{z}_{1:t-1}, \mathbf{f}_{1:t-1})}. \tag{3.10}$$

Proof. To obtain an inductive argument, suppose that at $t-1$ the hazard estimate $\psi_{t-1}(H) = p(H \mid \mathbf{z}_{1:t-1}, \mathbf{f}_{1:t-1}) = p(H \mid \mathbf{f}_{1:t-1})$ is independent of the sensor measurements $\mathbf{z}_{1:t-1}$. Then the recursive Bayesian filter update for time t gives

$$\psi_t(H) = \frac{p(\mathbf{z}_t, \mathbf{f}_t \mid H)\psi_{t-1}(H)}{\sum_{H \in \mathcal{H}} p(\mathbf{z}_t, \mathbf{f}_t \mid H)\psi_{t-1}(H)}.$$

Now assuming that H and S are not deterministically related,

$$p(\mathbf{z}_t, \mathbf{f}_t \mid H) = p(\mathbf{z}_t \mid \mathbf{f}_t, H)p(\mathbf{f}_t \mid H) = p(\mathbf{z}_t \mid \mathbf{f}_t)p(\mathbf{f}_t \mid H),$$

where the last equality exists because the measurement, $\mathbf{z}_t$, is independent of the hazards, H, given the failure, $\mathbf{f}_t$, as described in the previous section. This leads to

$$\psi_t(H) = \frac{p(\mathbf{z}_t \mid \mathbf{f}_t)p(\mathbf{f}_t \mid H)\psi_{t-1}(H)}{p(\mathbf{z}_t \mid \mathbf{f}_t)\sum_{H \in \mathcal{H}} p(\mathbf{f}_t \mid H)\psi_{t-1}(H)},$$

and the factor of $p(\mathbf{z}_t \mid \mathbf{f}_t)$ cancels in the numerator and denominator to obtain (3.9). Now notice that $\psi_t(H) = p(H \mid \mathbf{z}_{1:t}, \mathbf{f}_{1:t}) = p(H \mid \mathbf{f}_{1:t})$ remains independent of the measurements at time t. The initial distribution, $\psi_0(H)$, must be independent of $\mathbf{z}_{1:t}$ (because no measurements have been collected yet), therefore by mathematical induction the hazard estimate distribution conditioned on the failures is always independent of the measurements.

Using a similar mathematical induction argument, suppose that the hazard and event estimates are independent given the measurements and failures up to time $t-1$, so $p(H, X \mid \mathbf{z}_{1:t-1}, \mathbf{f}_{1:t-1}) = \varphi_{t-1}(X)\psi_{t-1}(H)$. Then the Bayesian update for their joint distribution at time t is given by

$$p(X, H \mid \mathbf{z}_{1:t}, \mathbf{f}_{1:t}) = \frac{p(\mathbf{z}_t, \mathbf{f}_t \mid X, H)\varphi_{t-1}(X)\psi_{t-1}(H)}{\sum_{X\in\mathcal{X}}\sum_{H\in\mathcal{H}} p(\mathbf{z}_t, \mathbf{f}_t \mid X, H)\varphi_{t-1}(X)\psi_{t-1}(H)}.$$

Factoring the numerator using the conditional independence of the measurements and hazards given the failures,

$$p(X, H \mid \mathbf{z}_{1:t}, \mathbf{f}_{1:t}) = \frac{p(\mathbf{z}_t \mid \mathbf{f}_t, X)p(\mathbf{f}_t \mid H)\varphi_{t-1}(X)\psi_{t-1}(H)}{\sum_{X\in\mathcal{X}}\sum_{H\in\mathcal{H}} p(\mathbf{z}_t \mid \mathbf{f}_t, X)p(\mathbf{f}_t \mid H)\varphi_{t-1}(X)\psi_{t-1}(H)},$$

and separating terms that depend on X from those that depend on H yields

$$p(X, H \mid \mathbf{z}_{1:t}, \mathbf{f}_{1:t}) = \frac{p(\mathbf{z}_t \mid \mathbf{f}_t, X)\varphi_{t-1}(X)}{\sum_{X\in\mathcal{X}} p(\mathbf{z}_t \mid \mathbf{f}_t, X)\varphi_{t-1}(X)} \frac{p(\mathbf{f}_t \mid H)\psi_{t-1}(H)}{\sum_{H\in\mathcal{H}} p(\mathbf{f}_t \mid H)\psi_{t-1}(H)}.$$

The right-most fraction is the Bayesian update from (3.9), and the left-most expression can be factored as $p(X, H \mid \mathbf{z}_{1:t}, \mathbf{f}_{1:t}) = p(X \mid H, \mathbf{z}_{1:t}, \mathbf{f}_{1:t})\psi_t(H)$, which gives

$$p(X \mid H, \mathbf{z}_{1:t}, \mathbf{f}_{1:t})\psi_t(H) = \frac{p(\mathbf{z}_t \mid \mathbf{f}_t, X)\varphi_{t-1}(S)}{\sum_{X\in\mathcal{X}} p(\mathbf{z}_t \mid \mathbf{f}_t, X)\varphi_{t-1}(X)}\psi_t(H).$$

The fraction on the right is independent of H, so $p(X \mid H, \mathbf{z}_{1:t}, \mathbf{f}_{1:t}) = p(X \mid \mathbf{z}_{1:t}, \mathbf{f}_{1:t}) = \varphi_t(X)$, which results in the Bayesian update in (3.8). Therefore if the estimate distributions of X and H are independent at time $t-1$ they will also be so at time t, and by induction, if their initial distributions are independent then they will remain so for all time.

Finally, in the case that the hazards and the events are deterministically related, the standard recursive Bayesian filter yields

$$p(X \mid \mathbf{z}_{1:t}, \mathbf{f}_{1:t}) = \frac{p(\mathbf{z}_t, \mathbf{f}_t \mid X) p(X \mid \mathbf{z}_{1:t-1}, \mathbf{f}_{1:t-1})}{\sum_{X \in \mathcal{X}} p(\mathbf{z}_t, \mathbf{f}_t \mid X) p(X \mid \mathbf{z}_{1:t-1}, \mathbf{f}_{1:t-1})},$$

which factors to the expression in (3.10). □

3.2.1 Decentralized Estimation

These filters may be decentralized, separating them into updates over individual measurements. This way each robot maintains a separate filter and is able to incorporate past measurements. Furthermore, this iterative update reduces the complexity of the Bayesian update to be linear, rather than exponential, in the number of robots. Note that since the environment is static (*i.e.*, the belief will not change if the robots cease taking measurements) and robots detections and failures are conditionally independent given the target and hazard locations, the current belief can be written as

$$\phi_t(X) \propto \prod_i \prod_{t=1}^{\tau_i} p(z_t^i \mid f_t^i, X, \mathbf{q}^i) \phi_0(X). \tag{3.11}$$

Thus the filtering approach described in Algorithm 1 will result in the same posterior regardless of the order in which individual updates are applied. These updates using individual measurements leverage the fact that each robot only sees a subset of the environment.

Theorem 2. *The Bayesian update over the full environment can be computed from the Bayesian update over the neighborhood $\mathcal{V}$ as*

$$\varphi_t(X) = \frac{\varphi_t(V)}{\varphi_{t-1}(V)} \varphi_{t-1}(X). \tag{3.12}$$

Proof. Begin by noting that (3.8) depends only upon the prior estimate and the detection likelihood (3.2). Since a sensor is only able to detect targets inside of its footprint, $p(z^i = 0 \mid f^i = 0, j) = 1$ for all $j \in X \setminus F^i$ so that $p(z^i = 0 \mid f^i = 0, X) = p(z^i = 0 \mid f^i = 0, r_T^i(X))$.

Since multiple X map to the same V,

$$\varphi(V) = \sum_{X|r_T^i(X)=V} \varphi(X). \tag{3.13}$$

Then the denominator of (3.8) is equal to $\sum_{\mathcal{V}} p(z_t^i \mid f_t^i, V^i)\varphi_{t-1}(V)$. Finally, using the relationship with $\varphi(V)$, the update equation becomes

$$\begin{aligned}
\varphi_t(X) &= \frac{p(z_t \mid f_t, X)\varphi_{t-1}(X)}{\sum_{\mathcal{X}} p(z_t \mid f_t, X)\varphi_{t-1}(X)} \frac{\varphi_{t-1}(V)}{\varphi_{t-1}(V)} \\
&= \frac{p(z_t \mid f_t, V)\varphi_{t-1}(V)}{\sum_{\mathcal{X}} p(z_t \mid f_t, V)\varphi_{t-1}(V)} \frac{\varphi_{t-1}(X)}{\varphi_{t-1}(V)} \\
&= \varphi_t(V) \frac{\varphi_{t-1}(X)}{\varphi_{t-1}(V)}.
\end{aligned}$$

□

The hazard updates can be similarly decomposed using the projection $r_H^i : \mathcal{H} \to \mathcal{W}$. Statistics of interest of these distributions include the probability of n targets in the environment, $p(|X| = n) = \sum_{X||X|=n} \varphi(X)$, and, as a special case of (3.13), the probability of an individual cell i being occupied,

$$\varphi(i \in X) = \sum_{X|i \in X} \varphi(X). \tag{3.14}$$

3.2.2 Adaptive Cellular Decomposition

In order to store the full distribution over RFSs for a large-scale environment, the total number of cells used must be kept at a tractable level. This section outlines one solution to this problem, using an adaptive cell decomposition based on the quadtree data structure. However, the methodology can be easily extended to work with other decompositions. Quadtrees have been used in other localization tasks, such as in the work of Carpin et al. [8]. What distinguishes this approach is the use of finite set statistics and the ability to remove leaves from the tree.

The main idea is that initially a coarse discretization is used, which is refined only in areas that are likely to contain an object. If the detection turns out to be a false positve, the procedure can then be reversed. The two basic operations of this adaptive cellular decomposition are the addition and removal of a cell, given in Algorithm 2 and Algorithm 3, respectively. In Algorithm 3, $r_j(X) : \mathcal{X} \to \mathcal{V}_{j^{\complement}}$ is a projection onto the complement of cell j, $j^{\complement}$.

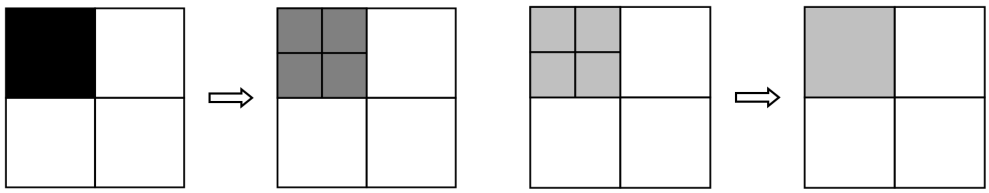
Algorithm 2 Add Cell

1: $\mathcal{X}' \leftarrow \mathcal{X}$
2: **for** $X \in \mathcal{X} \mid |X| <$ max number of targets **do**
3: $\quad \varphi'(X) \leftarrow \frac{1}{2}\varphi(X)$
4: $\quad \varphi'(X \cup \{m_T + 1\}) \leftarrow \frac{1}{2}\varphi(X)$
5: $\quad \mathcal{X}' \leftarrow \mathcal{X}' \cup \{X \cup \{m_T + 1\}\}$
6: **end for**
7: $m'_T \leftarrow m_T + 1$

Algorithm 3 Remove Cell

1: **for** $V \in \mathcal{V}_{j^{\complement}}$ **do**
2: $\quad \varphi(V) \leftarrow \sum_{X|r_j(X)=V} \varphi(X)$
3: $\quad \mathcal{X}' \leftarrow \mathcal{X}' \setminus \{V \cup \{j\}\}$
4: **end for**
5: $m_T \leftarrow m_T - 1$

These two operations can then be used to adapt any cellular decomposition of the environment. A refinement procedure involves removing cells that are occupied with sufficiently high probability, $\varphi(i \in X) > \tau_o$ for some threshold τ_o, and then adding some number of child cells, four in the case of a quadtree. This is illustrated in Figure 4a. Similarly the cell merging procedure, illustrated in Figure 4b, takes place if all children of a single parent are occupied with sufficiently low probability, *i.e.*, $\varphi(j \in X) < \tau_e < \tau_o$ for all children j of a single parent cell. All children are removed, and then the parent is added. Note that the distribution used in the Bayesian filter is over RFSs, which is then used to calculate the occupancy probability of individual cells used in the grid adaptation via (3.14). These cell addition and removal operations preserve the probability that the parent cell is occupied.



(a) Cell refinement (b) Cell merge

Figure 4: A simple 2×2 grid example where the shading indicates the probability that a cell is occupied with white being 0 and black being 1. A cell refinement procedure is shown in (a), where a large occupied cell is divided into four smaller cells with uniform occupancy probability. A grid merging procedure is shown in (b), where four empty sub-cells with the same parent cell are merged to form the parent cell.

Analytic Bound on Error in Likelihoods

Objects can be localized with arbitrary precision (by picking a minimum cell size) while simultaneously keeping the total number of cells low by using this adaptive approach. In fact, under mild assumptions about the choices of the g_s, g_h functions from equations (3.3) to (3.4), it is possible to show that the error between the true feature detection likelihoods and the approximations is bounded and goes to zero as the cell size approaches zero.

Whatever the choice of g_s, g_h, it is a reasonable assumption that the functions will be monotonically decreasing. Let the true target be at an arbitrary point $\mathbf{e} \in E_j^s$. Let $g_s(\|\mathbf{q}^i - \mathbf{e}\|) = g_s(\mathbf{q}^i, \mathbf{e})$ since the sensor is assumed to be isotropic and let ℓ be the diameter of the cell E_j^s which has points $\{\mathbf{e}_{j,k}^s\}_{k=1}^m$. Then the error ε in the detection likelihood of a target at location $\mathbf{e} \in E_j^s$ will be

$$\varepsilon = \frac{1}{m} \sum_{k=1}^{m} \left[g_s(\|\mathbf{q}^i - \mathbf{e}_{j,k}^s\|) - g_s(\|\mathbf{q}^i - \mathbf{e}\|) \right]. \tag{3.15}$$

By the triangle inequality, $\|\mathbf{q}^i - \mathbf{e}\| \leq \|\mathbf{q}^i - \mathbf{e}_{j,k}^s\| + \|\mathbf{e}_{j,k}^s - \mathbf{e}\| \leq \|\mathbf{q}^i - \mathbf{e}_{j,k}^s\| + \ell$ for all k. Then by monotonicity of the function, $g_s(\|\mathbf{q}^i - \mathbf{e}\|) \geq g(\|\mathbf{q}^i - \mathbf{e}_{j,k}^s\| + \ell)$, so

$$\varepsilon \leq \frac{1}{m} \sum_{k=1}^{m} \left[g_s(\|\mathbf{q}^i - \mathbf{e}_{j,k}^s\|) - g_s(\|\mathbf{q}^i - \mathbf{e}_{j,k}^s\| + \ell) \right]. \tag{3.16}$$

Similarly, $\|\mathbf{q}^i - \mathbf{e}\| \geq \|\mathbf{q}^i - \mathbf{e}^s_{j,k}\| - \ell$ so

$$\varepsilon \geq \frac{1}{m} \sum_{k=1}^{m} \left[g_s(\|\mathbf{q}^i - \mathbf{q}_{j,k}\|) - g_s(\|\mathbf{q}^i - \mathbf{q}_k\| - \ell) \right] \tag{3.17}$$

where $g_s(r) = g_s(0)$ for all $r \leq 0$ so that g_s is defined over all $\mathbb{R}$.

Looking at these bounds there are two limiting cases. The first is when the cells are small, i.e. $\ell \to 0$. In this case $\|\mathbf{q}^i - \mathbf{e}^s_{j,k}\| \pm \ell \to \|\mathbf{q}^i - \mathbf{e}^s_{j,k}\|$ so $\varepsilon \to 0$. The second case is when the detection likelihood is sufficiently flat so all points in the cell are equivalent from a sensing standpoint, or that $g_s(\|\mathbf{q}^i - \mathbf{e}^s_{j,k}\| \pm \ell) \approx g_s(\|\mathbf{q}^i - \mathbf{e}^s_{j,k}\|)$. In fact, with the finite footprint assumption, $g_s(r) = 0$ for all r greater than the sensing radius R, so the error trivially will be zero outside of this radius. The same reasoning holds for the function g_h.

Multiple Sensor Modalities

The idea of an adaptive cell decomposition also implies an adaptive sensing strategy. When the grid is coarse then a coarse sensor (*i.e.*, having a large radius but relatively noisy) can be used and when the grid is finer then the robots can switch to a finer sensor (*i.e.*, with low noise but a small radius). For example, in a search and rescue mission where the team is searching for victims trapped in a building, robots could be equipped with an audio sensor which would allow the robots to hear somebody calling for help in a large radius around them, and the robots could then switch to a camera when they approached a person.

3.3 Mutual Information Gradient Controller

This section derives an information seeking controller using the analytic gradient of mutual information, originally published in [90]. The mutual information of two random variables is an information theoretic quantity [20, 91] that describes the amount of information that can be gained about one random variable (*e.g.*, targets) by observing another (*e.g.*, measurements). Mutual information is defined as

$$I[\mathcal{X}; \mathcal{Z}] = \int_{\mathcal{X}} \int_{\mathcal{Z}} p(X, \mathbf{z}) \log \frac{p(X, \mathbf{z})}{p(X)p(\mathbf{z})} \, d\mathbf{z} \, \delta X. \tag{3.18}$$

Here the information is written as though $\mathcal{X}, \mathcal{Z}$ were continuous random variables, however equivalent expressions can be written for the discrete case. The integral has also been replaced by a set integral as we have a distribution over random finite sets. The key to deriving the controller is to note that this information depends upon the locations of the robots through the failure and detection models. Then the gradient of mutual information with respect to the parameter q is given in the following Theorem.

Theorem 3. *Let random vector $\mathcal{Z}$ and random finite set $\mathcal{X}$ be jointly distributed with distribution $p(\mathcal{X}, \mathcal{Z} \mid \mathbf{q})$ that is differentiable with respect to the parameter vector $\mathbf{q} \in \mathbb{R}^{2n}$ over $E^n \subset \mathbb{R}^{2n}$. Also, suppose that the support $\mathcal{X} \times \mathcal{Z}$ of $p(\mathcal{X}, \mathcal{Z} \mid \mathbf{q})$ does not depend on $\mathbf{q}$. Then the gradient of mutual information with respect to the parameters $\mathbf{q}$ over E^n is*

$$\frac{\partial I[\mathcal{X}; \mathcal{Z} \mid \mathbf{q}]}{\partial \mathbf{q}} = \iint_{\mathcal{Z},\mathcal{X}} \frac{\partial p(X, \mathbf{z} \mid \mathbf{q})}{\partial \mathbf{q}} \log \frac{p(X, \mathbf{z} \mid \mathbf{q})}{p(X)p(\mathbf{z} \mid \mathbf{q})} \, \delta X d\mathbf{z}. \tag{3.19}$$

Proof. The theorem follows straightforwardly by applying the rules of differentiation. Notably, several terms are identically zero, yielding the simple result. First, the differentiation is moved inside the integrals since $\mathcal{X}$ and $\mathcal{Z}$ do not depend on the parameters $\mathbf{q}$. Then applying the chain rule to the integrand results in

$$\begin{aligned}\frac{\partial I[\mathcal{X}; \mathcal{Z} \mid \mathbf{q}]}{\partial \mathbf{q}} = &\int_{\mathbf{z} \in \mathcal{Z}} \int_{X \in \mathcal{X}} \frac{\partial p(X, \mathbf{z})}{\partial \mathbf{q}} \log \frac{p(X, \mathbf{z})}{p(X)p(\mathbf{z})} \, \delta X \, d\mathbf{z} + \int_{\mathbf{z} \in \mathcal{Z}} \int_{X \in \mathcal{X}} p(X, \mathbf{z}) \frac{p(X)p(\mathbf{z})}{p(X, \mathbf{z})} \\ &\times \Big[\frac{\partial p(X, \mathbf{z})}{\partial \mathbf{q}} \frac{1}{p(X)p(\mathbf{z})} - \frac{\partial p(X)}{\partial \mathbf{q}} \frac{p(\mathbf{z})p(X, \mathbf{z})}{(p(X)p(\mathbf{z}))^2} - \frac{\partial p(\mathbf{z})}{\partial \mathbf{q}} \frac{p(X)p(X, \mathbf{z})}{(p(X)p(\mathbf{z}))^2} \Big] \, \delta X \, d\mathbf{z},\end{aligned}$$

where the dependence on $\mathbf{q}$ is suppressed to simplify notation. Bringing $1/(p(X)p(\mathbf{z}))$ in front of the brackets gives

$$\begin{aligned}\frac{\partial I[\mathcal{X}; \mathcal{Z} \mid \mathbf{q}]}{\partial \mathbf{q}} = &\int_{\mathbf{z} \in \mathcal{Z}} \int_{X \in \mathcal{X}} \frac{\partial p(X, \mathbf{z})}{\partial \mathbf{q}} \log \frac{p(X, \mathbf{z})}{p(X)p(\mathbf{z})} \, \delta X \, d\mathbf{z} \\ &+ \int_{\mathbf{z} \in \mathcal{Z}} \int_{X \in \mathcal{X}} \Big[\frac{\partial p(X, \mathbf{z})}{\partial \mathbf{q}} - \frac{\partial p(X)}{\partial \mathbf{q}} p(\mathbf{z} \mid X) - \frac{\partial p(\mathbf{z})}{\partial \mathbf{q}} p(X \mid \mathbf{z}) \Big] \, \delta X \, d\mathbf{z}.\end{aligned}$$

Consider the three terms in the second double integral. For the first term,

$$\int_{\mathbf{z}\in\mathcal{Z}}\int_{X\in\mathcal{X}} \frac{\partial p(X,\mathbf{z})}{\partial \mathbf{q}}\,\delta X\,d\mathbf{z} = \frac{\partial}{\partial \mathbf{q}}\int_{\mathbf{z}\in\mathcal{Z}}\int_{X\in\mathcal{X}} p(X,\mathbf{z})\,\delta X\,d\mathbf{z} = \frac{\partial}{\partial \mathbf{q}} 1 = 0.$$

For the second term,

$$\int_{\mathbf{z}\in\mathcal{Z}}\int_{X\in\mathcal{X}} \frac{\partial p(X)}{\partial \mathbf{q}} p(\mathbf{z}\mid X)\,\delta X\,d\mathbf{z}$$
$$= \int_{X\in\mathcal{X}} \frac{\partial p(X)}{\partial \mathbf{q}}\Big(\int_{\mathbf{z}\in\mathcal{Z}} p(\mathbf{z}\mid X)\,d\mathbf{z}\Big)\,\delta X = \frac{\partial}{\partial \mathbf{q}}\int_{X\in\mathcal{X}} p(X)\,\delta X = 0,$$

and the third term follows by interchanging $\mathbf{z}$ and X. □

Remark 1. The result holds for the general definition of mutual information and makes no assumptions as to the distribution of the random variables, or the form of the dependence of $p(X,\mathbf{z}\mid\mathbf{q})$ on its parameters. The result also holds for generally distributed random variables including discrete valued ones. The result is given for mutual information in "nats" (the base of the logarithm is e), but the same result holds with a multiplicative constant of $\log_2 e$ for mutual information in "bits."

Remark 2. It is interesting that the gradient of $I[\mathcal{X};\mathcal{Z}\mid\mathbf{q}]$ has the same form as $I[\mathcal{X};\mathcal{Z}\mid\mathbf{q}]$ itself, except that the first occurrence of $p(X,\mathbf{z}\mid\mathbf{q})$ is replaced by its gradient with respect to $\mathbf{q}$. This analytic expression for the gradient of mutual information has not been reported in the literature despite the proliferation of gradient based methods for maximizing mutual information in various applications ranging from channel coding [78, 79], to medical imaging alignment [100], to the control of robotic sensor networks [41]. In [79] the authors derive an expression that can be shown to be equivalent to a special case of Theorem 3 in which $p(X)$ is not a function of $\mathbf{q}$.

Remark 3. Arbitrary derivatives of mutual information can also be calculated, though the fortuitous cancellations do not necessarily hold for higher order derivatives. It would be, for example, to compute the Hessian of mutual information to examine the coupling between the control laws for neighboring robots.

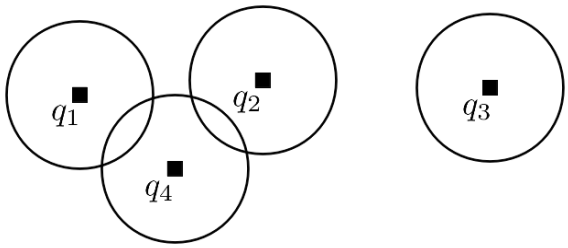


Figure 5: In this situation the team of robots, with the footprints of the individual robots shown by the circles, is divided into two cliques $C_1 = \{1, 2, 4\}, C_2 = \{3\}$.

3.3.1 Finite Footprint Approximation

The mutual information may be decoupled by leveraging the fact that sensors have a finite field of view. This can significantly reduce the complexity of the gradient calculations. Consider the example in Figure 5 where the fields of view of the robots overlap in different ways. Let $\mathcal{G}(Q, E)$ be an undirected graph, with a node q^i for each sensor. Edges are added between nodes if their sensor footprints overlap, *i.e.*, edge e_{ij} is added if $F^i \cap F^j \neq \varnothing,\ i \neq j$. In this example there are two edges, (1,4) and (2,4). A *clique* of sensors is a connected component of this graph, denoted C_i. Algorithm 4 presents a decentralized algorithm to compute these cliques. The use of a superscript ^{C_i} represents the union of that quantity over the clique C_i, for example the union of the sensor footprint is F^{C_i}, or the cross product over the robots in the clique, for example the set of all tuples of measurements is $\mathcal{Z}^{C_i}$.

Algorithm 4 Clique Identification For Robot i

$Q^i_k = \{\mathbf{q}^j_t \mid \|\mathbf{q}^j_t - \mathbf{q}^i_t\| \leq R_c, F^i \cap F^j \neq \varnothing\}$
$k = 0$
repeat
    $k \leftarrow k + 1$
    $Q^i_k \leftarrow Q^i_{k-1}$
    for $j \mid \|\mathbf{q}^j_t - \mathbf{q}^i_t\| \leq R_c$ **do**
        $Q^i_k \leftarrow Q^i_k \cup Q^j_k$
    end for
until $Q^i_k = Q^i_{k-1}$

Using the the chain rule for mutual information given by Cover and Thomas [20], the

expression for mutual information becomes

$$I[\mathcal{X};\mathcal{Z}] = \sum_i I[\mathcal{X};\mathcal{Z}^{C_i} \mid \mathcal{Z}^{C_1},\ldots,\mathcal{Z}^{C_{i-1}}]. \tag{3.20}$$

Recall that measurements of different agents are conditionally independent given the locations of the targets. However, the measurements of two robots in different cliques are, in general, dependent. To see this, consider the situation where there are two robots searching for a single target. If one robot receives a detection then the other robot is less likely to, even if the sensor footprints the robots are completely disjoint. Mathematically, the measurements become dependent after marginalizing out over all possible target positions,

$$p(z^1, z^2) = \int p(z^1 \mid \mathbf{x}) p(z^2 \mid \mathbf{x}) p(\mathbf{x})\, d\mathbf{x}.$$

This same logic holds for the multi-target case, since the positions of targets may be correlated.

However, assuming that the correlation between inter-clique measurements is significantly smaller than with intra-clique measurements, mutual information becomes

$$I[\mathcal{X};\mathcal{Z} \mid Q] \approx \sum_i I[\mathcal{X};\mathcal{Z}^{C_i} \mid Q^{C_i}] = \sum_i I[\mathcal{V}^{C_i};\mathcal{Z}^{C_i} \mid Q^{C_i}]. \tag{3.21}$$

Note that the second equality is exact, as measurements by a clique of sensors are independent of targets outside the clique's footprint. The error in this approximation is

$$I[\mathcal{X};\mathcal{Z} \mid Q] - \sum_i I[\mathcal{V}^{C_i};\mathcal{Z}^{C_i} \mid Q^{C_i}] = H[\mathcal{Z} \mid Q] - \sum_i H[\mathcal{Z}^{C_i} \mid Q^{C_i}], \tag{3.22}$$

which is bounded from above by zero with equality iff the measurements are independent, as shown by Cover and Thomas [20].

Similarly, the gradient in (3.19) can be written as

$$\frac{\partial I[\mathcal{X};\mathcal{Z} \mid \mathbf{q}]}{\partial \mathbf{q}} \approx \sum_i \frac{\partial I[\mathcal{V}^{C_i};\mathcal{Z}^{C_i} \mid \mathbf{q}]}{\partial \mathbf{q}}. \tag{3.23}$$

Note that the gradient of mutual information with respect to the position of each sensor depends only upon the locations of the sensors in its clique.

The intuition behind this is that robots near each other should coordinate their motions to better localize targets while robots that are sufficiently far apart can act as independent agents with little penalty to performance. Using this reasoning, it is simple to adapt the definition of a clique to fit the computational budget of a particular robot: coordinate with robots that are increasingly separated until a maximum number of robots is reached or no other robots are within communication range. This approach is similar in spirit to the single- and pairwise-node approximations presented by Hoffmann and Tomlin [45], where mutual information is approximated by the sum of mutual information from each sensor or each pair of sensors in the network. However this approach offers a systematic method for how to best spend the computational resources available to each robot.

3.3.2 Control Law

Writing the gradient from (3.19) in terms of known quantities,

$$\frac{\partial I[\mathcal{V}^C; \mathcal{Z}^C \mid Q^C]}{\partial \mathbf{q}} = \sum_{\mathbf{z}^C \in \mathcal{Z}^C} \sum_{V \in \mathcal{V}^C} \sum_{W \in \mathcal{W}^C} \frac{\partial p(\mathbf{z}^C \mid V, W, Q^C)}{\partial \mathbf{q}} \varphi_t(V) \psi_t(W) \\ \times \log \frac{\sum_{W \in \mathcal{W}^C} p(\mathbf{z}^C \mid V, W, Q^C) \psi_t(W)}{\sum_{V \in \mathcal{V}^C} \sum_{W \in \mathcal{W}^C} p(\mathbf{z}^C \mid V, W, Q^C) \varphi_t(V) \psi_t(W)}. \quad (3.24)$$

Here $\varphi_t(V)$ and $\psi_t(W)$ come from (3.8) and (3.9) and $p(z^C \mid V, W, q^C)$ from (3.7). The gradient of (3.2) for a single robot is

$$\frac{\partial p(z^i = 0 \mid V, W, \mathbf{q})}{\partial \mathbf{q}} = -p(f^i = 0 \mid W, \mathbf{q}) p(z^i = 0 \mid V, W, \mathbf{q}) \times \sum_{j \in V} \frac{1}{1 - \mu(\mathbf{q}, E_j^s)} \frac{\partial \mu(\mathbf{q}, E_j^s)}{\partial \mathbf{q}} \\ + p(f^i = 0 \mid W, \mathbf{q}) p(z^i = 1 \mid V, W, \mathbf{q}) \times \sum_{k \in W} \frac{1}{1 - \alpha(\mathbf{q}, E_k^h)} \frac{\partial \alpha(\mathbf{q}, E_k^h)}{\partial \mathbf{q}}, \quad (3.25)$$

and when $z_i = 1$ it is simply the negative of this. The derivative of (3.7) is found using (3.25) and the chain rule.

Using these results, the proposed controller is

$$\mathbf{q}_{t+1}^i = \mathbf{q}_t^i + k \frac{\frac{\partial I[\mathcal{V}^C; \mathcal{Z}^C | q_t]}{\partial \mathbf{q}_t^i}}{\left\| \frac{\partial I[\mathcal{V}^C; \mathcal{Z}^C | \mathbf{q}_t]}{\partial \mathbf{q}_t^i} \right\| + \epsilon}, \tag{3.26}$$

where $i \in C$, k is the maximum step size, and $\epsilon \ll 1$ avoids singularities near critical points. It is important to note that this is not a traditional gradient ascent controller, since mutual information changes as measurements and failures are incorporated into the target and hazard beliefs. Also, hazard avoidance is implicitly built into the proposed controller since the information gained by a failed sensor is zero so that robots naturally avoid areas where they expect to fail.

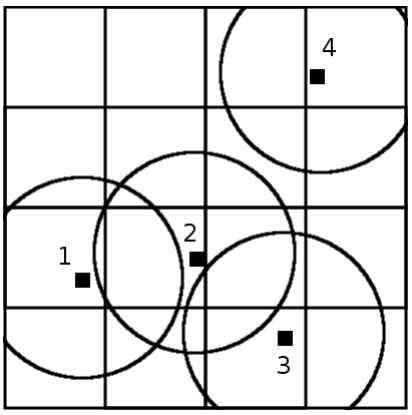
3.3.3 Computational Complexity

As can be seen from (3.24), the number of computations involved in the mutual information gradient for a single robot in clique C is $O(2^{|C|}|\mathcal{V}^C||\mathcal{W}^C|)$. This is exponential in the number of robots in a clique and linear in the number of possible RFSs in the clique footprint. Both depend on the size of the footprint and maximum number of targets, specifically

$$|\mathcal{V}^C| = \sum_{k=0}^{|X|_{\max}} \binom{|F^C|}{k}, \tag{3.27}$$

where $|F^C|$ is the number of cells within the union of the sensor footprints for clique C and $|X_{\max}|$ is the maximum number of targets in the environment. So $|\mathcal{V}^C|$ is $O(|F^C|^{|X|_{\max}})$ when $|X|_{\max} \ll |F^C|$ and $O(2^{|F^C|})$ when $|X|_{\max} \approx |F^C|$. $|\mathcal{W}_C|$ scales analogously.

While the exact complexity depends upon the current configuration of the team of robots as well as the representation of the environment, we can examine two limiting cases. The complexity is lower-bounded when each robot is in its own clique, which is $O(|\mathcal{V}^i||\mathcal{W}^i|)$. On the other extreme, the complexity is upper-bounded when all robots are in a single clique, in which case it is $O(2^n|\mathcal{V}^C||\mathcal{W}^C|)$. Despite the lack of guarantee of reduced complexity, empirically the performance increases when using the sensor grouping. The computational benefits will increase as the size of the environment increases because it is more likely for



(a) Test Environment

Method	Cliques For Robot 1
Full	$\{1,2,3,4\}$
SN	$\{1\}$
PW	$\{1\},\{1,2\},\{1,3\},\{1,4\}$
FI	$\{1,2\}$
FF	$\{1,2,3\}$

(b) Cliques for Robot 1

Figure 6: The locations of the robots in the test environment, given by the black squares with their sensor footprints indicated by the circles, are shown on the left. The cliques containing robot 1 are shown in the table on the right for each approximation method.

robots to split into separate cliques.

3.4 Multi-Robot Simulation and Results

Looking at the error due to the finite footprint approximation given in (3.22), it is difficult to gain intuition as to how tight of a bound this is and how much the approximation affects the control decision. Also, as discussed in Section 3.2.2, the computational complexity depends upon the relative configuration of the robot team and is thus also difficult to quantify. A series of simulations will illustrate the performance of several approximations with respect to the full mutual information:

- Single-node (SN): robots act independently $O(|\mathcal{V}^i||\mathcal{W}^i|)$
- Pairwise (PW): robots consider pairwise information with each other robot $O(\sum_{j\neq i}|\mathcal{V}^i\cup\mathcal{V}^j||\mathcal{W}^i\cup\mathcal{W}^j|)$
- Footprint intersection (FI): robots consider only other robots whose footprints intersect their own (*i.e.*, redefine cliques from Section 3.3.1 to be neighbors in the graph)
- Finite-footprint (FF): described in Section 3.3.1

The SN and PW approximations were proposed by Hoffmann and Tomlin [45].

Table 3: Comparison of approximation methods for uniform belief.

Method	Full	SN	PW	FI	FF
Time (s)	0.2061	0.0201	0.0705	0.0217	0.0346
Mean Error (°)	–	-10.848	-0.500	-0.501	0.492
Max Error (°)	–	136.1	-115.2	-84.95	-65.3

For the comparative test, a team of robots begins in the configuration specified in Figure 6a. The control direction computed for robot 1 is compared across all planning approximations. This configuration is chosen so that the robot under consideration belongs to a different clique (or set of cliques) for each approximation method, as listed in Table 3. This table also contains the mean computational time for each method. As is expected, the full computation takes considerably longer than any of the approximations, with the approximation presented in Section 3.3.1 performing favorably with respect to existing approximation techniques.

As can be seen from (3.24), the direction of the gradient depends upon the current belief about the environment. Consider two cases: one with a near-uniform distribution and one with a highly peaked distribution. The first case is a near uniform distribution, with small random perturbations. Random distributions are generated by uniformly sampling a number in the range $[0, 1]$ for each RFS and then normalizing these to form a distribution. Robot 1 computes the gradient of mutual information using 500 such samples, using the exact computations and each of the four approximations from above. Table 3 shows the mean error in the direction, relative to the full mutual information computation. These errors are relatively small, except when using SN, and the results fairly consistent with standard deviations on the order of $0.1°$ for each approximation. This implies that there is little correlation between robots with non-overlapping sensor footprint when the uncertainty in the target position is high.

The second case is a highly peaked distribution. A distribution is initialized for each possible RFS containing at most five targets in the environment Figure 6a. These distributions have 95% of the probability mass on the particular RFS of interest and the remaining 5% of the mass is distributed uniformly across all other RFSs. Figure 7 shows box plots of

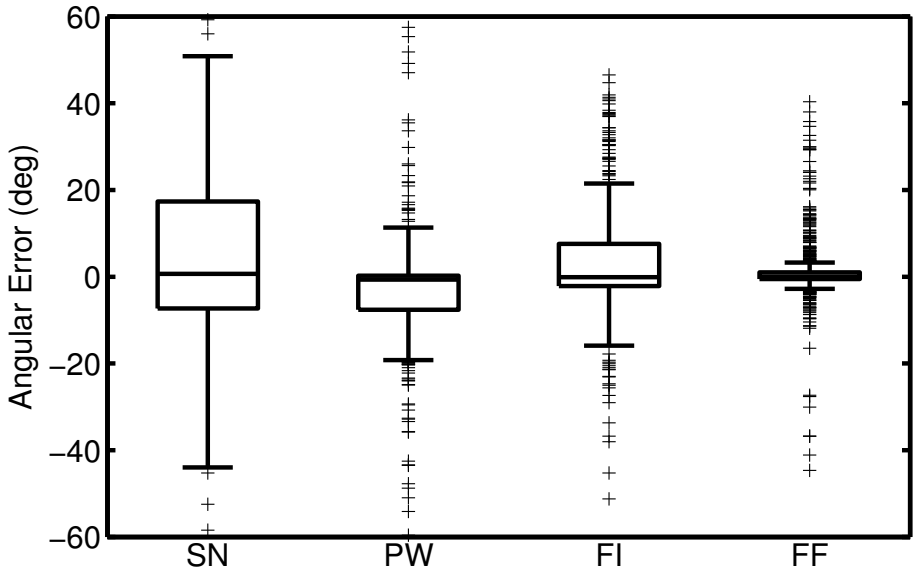


Figure 7: Box plots showing the error in angle of the gradient approximations for each of the approximation methods, measured in degrees. Each box plot represents ~ 7000 data points. Not shown are 15.8%, 1.80%, 0.08%, and 0.04% of the data, for the SN, PW, FI, and FF respectively, which correspond to large control deviations.

the error in the computed control direction for each approximation method. As the figure shows, SN performs poorly with a large spread in the direction error and relatively little probability mass near zero error. The PW and FI approximations perform comparably, with opposite biases in the distribution. PW is more peaked near zero error, but also has more outliers than FI. FF performs the best out of all four methods, with the most peaked distribution near zero error, the fewest outliers, and the best worst-case performance. Such large errors occur when the maximum likelihood target locations fall within the footprints of other robots in the group, in which case the decision made by robot 1 will be more strongly affected by the motion of the other robots in the team.

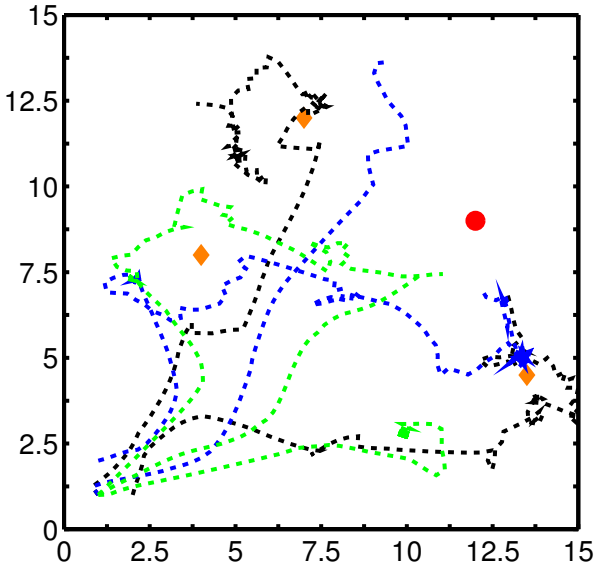
A series of simulations in the environment shown in Figure 8a test the performance of the proposed algorithm in terms of localizing targets. Figure 8a shows typical paths taken by the robots during exploration. The sensor model used is

$$g_s(\mathbf{q}^i, \mathbf{e}) = \begin{cases} (1 - p_{\text{fn}}) \exp\left(\frac{-\|\mathbf{q}^i - \mathbf{e}\|^2}{\sigma^2}\right), & \|\mathbf{q}^i - \mathbf{e}\| \leq R \\ 0, & \|\mathbf{q}^i - \mathbf{e}\| > R \end{cases} \tag{3.28}$$

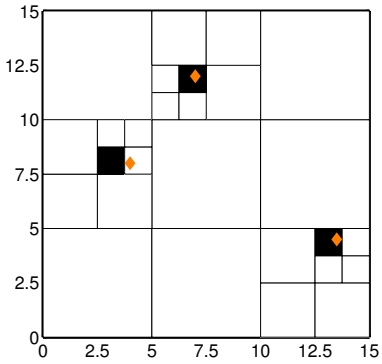
where $p_{\text{fn}} = 0.05, \sigma = 2$, and $R = 6$. The failure model g_h is of the same form, with $p_{\text{s}} = 0.1, \sigma = 1.5$, and $R = 2$. Figures 8b and 8d show the final estimates of the target and hazard RFSs, respectively, for the given path. The robots assume a maximum of four targets and two hazards in the environment. If a robot fails, a new one is sent out from a base station located at $(1, 1)$ in the environment. Figure 8c shows the entropy in the target location estimate. The entropy generally decreases, with increases due to the addition of cells as well as false positive and missed detections.

3.5 Ground Robot Experiments

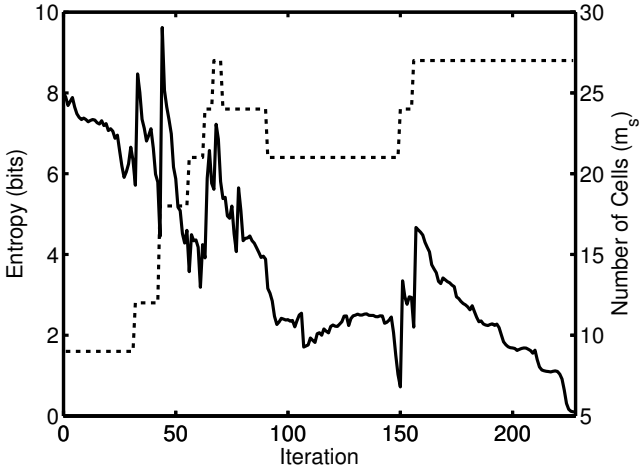
The ability for robots to locate and interact with objects of interest within an unstructured environment is highly important as robots move out of controlled settings. This section examines a prototypical example of playing fetch with a robot. First, the robot is shown a new object. Second, the robot must go into the field and locate a small, yet potentially unknown, number of these objects that are scattered within the environment using a monocular camera. Finally, after locating the objects, the robot collects them and returns them to the user. Such behavior has obvious extensions to household robots, inspection tasks, and search and rescue. Hardware experiments — with the robot shown in Figure 9 — validate the performance of the proposed exploration strategy in real-world scenarios. The platform is a differential drive robot built on a Segway platform with the maximum speed limited to 0.55 m/s. It is equipped with a single front-facing camera which detects objects using shape and color matching. There are also a pair of stereo cameras for visual odometry, a vertical-scanning LIDAR for pitch estimation to correct distance measurements, wheel encoders for odometry, and a horizontal scanning LIDAR for obstacle detection. Onboard processing is done using two Mac Mini computers running Ubuntu and the Robot Operating System (ROS) [1], each with 2.0 GHz Intel core i7 processors and 4 GB of RAM, mounted to the robot chassis. While the platform is outfitted with an extensive sensor suite, the front-facing color camera yields the best performance for object detection since the black-and-white stereos cameras are not as reliable and the LIDAR cannot detect small objects



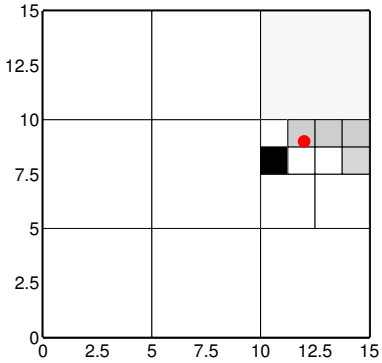
(a) Environment and robot paths



(b) Final target estimate



(c) Target entropy and number of cells



(d) Final hazard estimate

Figure 8: (a) Sample results in the trial environment. Target locations are given by the orange diamonds and hazard locations by the red square. (b,d) The shading in the cells represents $\varphi(j \in S), \psi(j \in H)$, where white is 0 and black is 1. (c) The solid line is the entropy of $\varphi(S)$ and the dashed line is the number of cells in the target discretization, m_s.

on the ground. The robot moves about in a bounded, planar environment, and the pose consists of its position and orientation.

3.5.1 Sensing

Monocular cameras provide more information than a simple, binary sensor. The camera (label 4 from Figure 9) observes a finite subset of the environment in front of the robot. The system used in these experiments uses a template matching algorithm, using shape and color information, to return the 2D positions of all known objects within the field of view. Das et al. [29] provide details on the vision subsystem and other components of the robot. The sensor returns a list of cells occupied by objects. The likelihood of such a measurement set is given by the expression from (2.26),

$$p(Z \mid X, \mathbf{q}) = e^{-\mu} \left(\prod_{\mathbf{z} \in Z} c(z) \right) \left(\prod_{\mathbf{x} \in X} 1 - p_d(\mathbf{x} \mid \mathbf{q}) \right) \times \left(\sum_{\theta} \prod_{j \mid \theta(j) \neq 0} \frac{p_d(\mathbf{x}_j \mid \mathbf{q}) g(\mathbf{z}_{\theta(j)} \mid \mathbf{x}_j, \mathbf{q})}{c(\mathbf{z}_{\theta(j)})(1 - p_d(\mathbf{x}_j \mid \mathbf{q}))} \right). \quad (3.29)$$

However, visual sensors can also be very noisy, returning false positives due to other objects in the environment (*e.g.*, if using shape detection to locate a ball, the wheel of a car is a potential false positive) and false negatives due to variable lighting conditions and occlusions. The detection, measurement, and clutter models are determined through a set of experiments. In these experiments, objects are placed at known locations the collects a sequence of measurements. Figure 10 shows the results of these experiments, overlaid on the sensor footprint. Below this is the detection likelihood function $p_d(\mathbf{x}_c \mid \mathbf{q})$, where $\mathbf{x}_c$ represents a continuous domain position. The single-target measurement model is the position of the target corrupted by Guassian noise, $g(\mathbf{z}_c \mid \mathbf{x}_c, \mathbf{q}) = x + \mathcal{N}(0, \Sigma)$, and the noise covariance Σ is found from this training data. The Gaussian is truncated after three standard deviations to ensure that the measurement model has compact support.

Due to the discrete representation of the environment, these continuous domain detection and measurement models must be converted into a discrete form. Assuming the target may

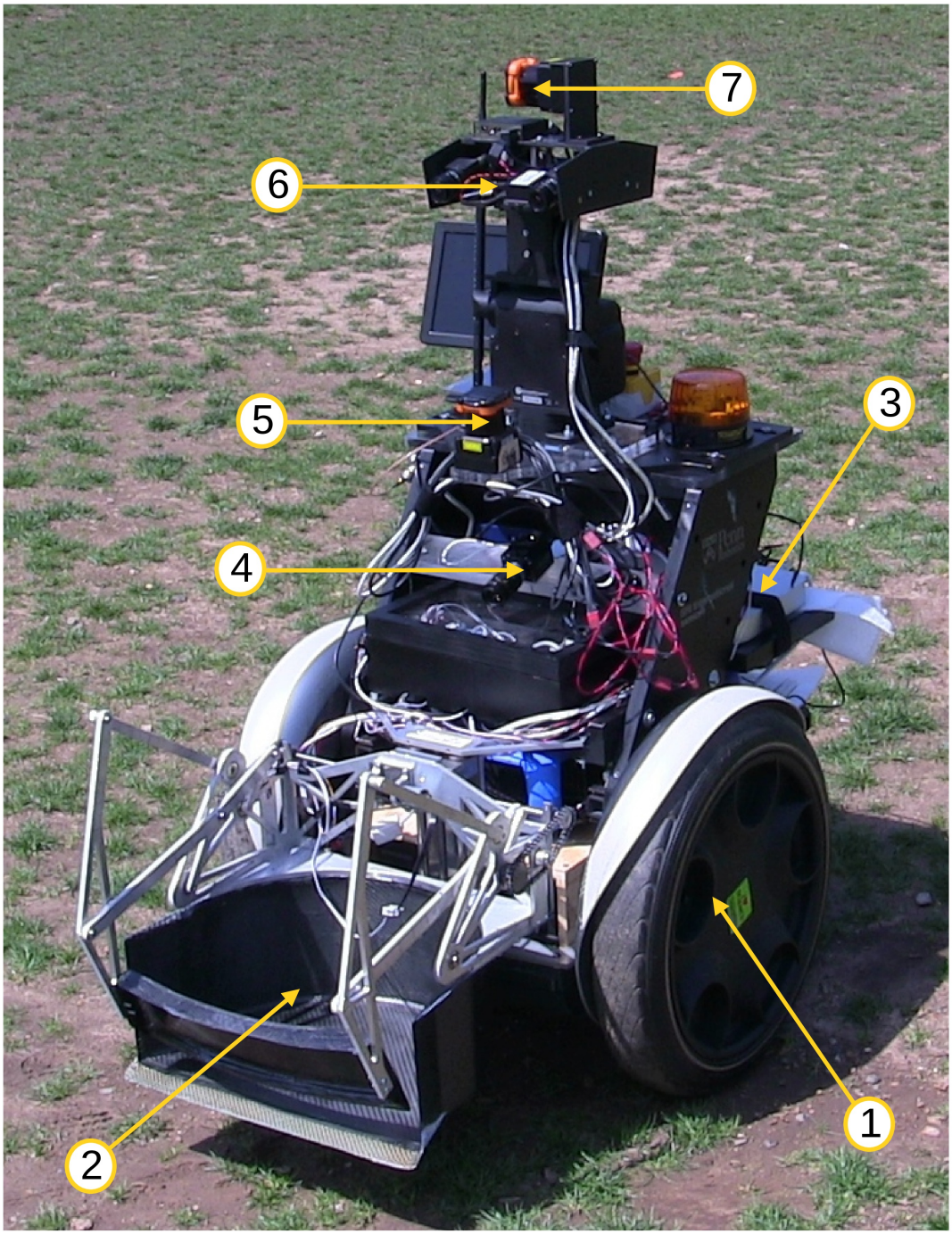


Figure 9: The Robot platform used in this work consisted of, 1) a Segway base, 2) an object scoop, 3) two Mac Minis, 4) a front-facing camera for object detection, 5) a horizontal-scanning LIDAR for obstacle avoidance, 6) a stereo camera for visual odometry, and 7) a vertical-scanning LIDAR for pitch estimation on uneven terrain.

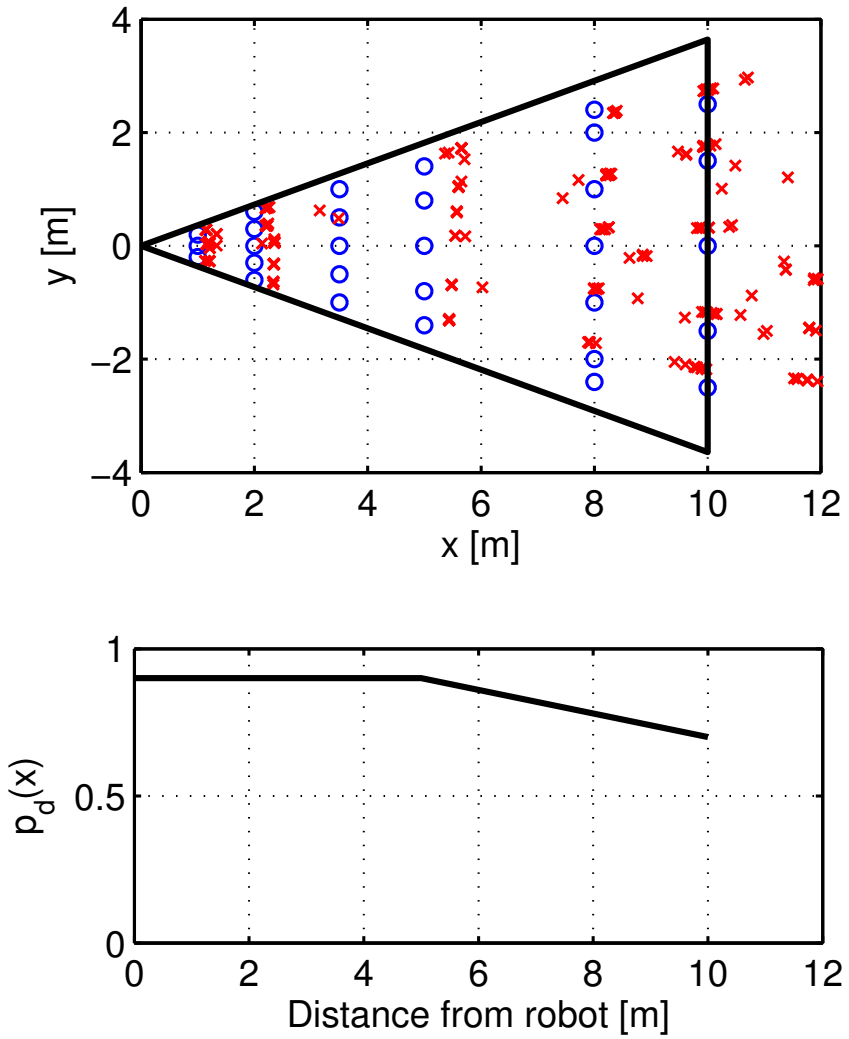


Figure 10: **Top**: Experimental results showing the true (blue circles) and estimated (red x's) object positions as measured in the body frame of the robot. This is superimposed on the sensor footprint and represents approximately 600 data points. **Bottom**: Detection likelihood as a function of the distance from the robot.

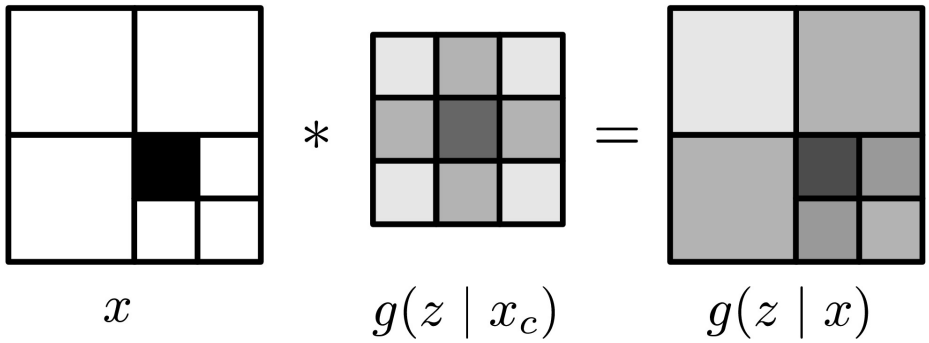


Figure 11: A simple 2×2 large cell, with uniform probability of the object being in each subcell, is convolved with the small cell measurement model. The resulting array can then be converted to the large cell model by simply taking the sum of the likelihood in each of the subcells. The relative sizes of each cell are to scale and the shading corresponds to the likelihood, with values outside the displayed cells being implicitly zero.

be located anywhere within the cell, the probability of detection is

$$p_d(\mathbf{x} \mid \mathbf{q}) = \frac{\int_{\mathbf{x}} p_d(\mathbf{x}_c \mid \mathbf{q})\, dx_c}{\int_{\mathbf{x}} d\mathbf{x}_c}, \tag{3.30}$$

As was done in (3.3) and (3.4), this integral may be approximated by a finite sum over representative points within the cell. This same process may be used to find the probability of a measurement originating from a target in a given cell (of the smallest resolution). This detection grid is then convolved with a candidate target cell, x, in a process similar to Gaussian blurring in image processing. Figure 11 illustrates this process. Note that this approach requires the small cell measurement model to have finite support. The support could be taken as the bounds of the environment if the sensor is able to see the entire environment.

The clutter is modeled as a Poisson random finite set with PHD $c(\mathbf{z})$. The expected number of clutter detections is $\mu = \int c(\mathbf{z})\, d\mathbf{z}$. Without any prior knowledge, the most natural choice is to set $c(\mathbf{z})$ to be uniform within the sensor footprint and zero outside, as no detections can occur outside the footprint. Based on experimental results, the expected number of clutter detections is set to $\mu = 0.05$.

3.5.2 Control

While the robot uses the full measurement sets to perform the filter updates, the control computations are prohibitively expensive. There are two main reasons for this: the number of possible target and measurements sets is very large, and for each target and measurement set pair, the robot must consider all possible data associations. Instead the robot considers the binary event $y = \mathbf{1}(|Z| = \varnothing)$. Either the robot sees at least one object within the footprint or it sees nothing. This reduces the integral over all possible measurement RFSs to a summation over the two possible outcomes of the binary event. This coarse model can be thought of as the probability of returning a "good" measurement, so maximizing this should lead to faster localization of the objects. This differs from the approach by Ristic and Vo [85], who use the full sensor model but sample from it to achieve computational tractability.

The controller uses a different detection model to that used in the estimation, as the piecewise-linear model in Figure 10 has a piecewise-constant gradient which always points directly backwards. The controller detection model, shown in Figure 12, is a truncated Gaussian in polar coordinates. This functional form was chosen for two primary reasons: it is differentiable everywhere except on the edge of the footprint, and it pushes the robot to center the camera (*i.e.*, the peak in the model) on regions of high uncertainty. The robot uses this new detection model in the control law (3.26). The experimentally derived probability of a false negative is $p_{\text{fn}} = 0.05$, the mean of the truncated Gaussian is at a range of 7 m directly in front of the robot, and the standard deviations are 2 m in the radial direction and 0.5 rad in the angular direction. The sensor has a 10 m range and a 40° field of view.

However, rather than differentiating mutual information with respect to its pose, the robot uses a virtual point at the center of the sensor's field of view. This virtual point is at the center of the peak of the binary sensor model in Figure 12. The robot then moves in such a way that the virtual point follows this gradient direction, effectively directing the sensor field of view towards regions of high uncertainty. Note that the detection model monotonically decreases with the distance from this virtual point, allowing us to bound the

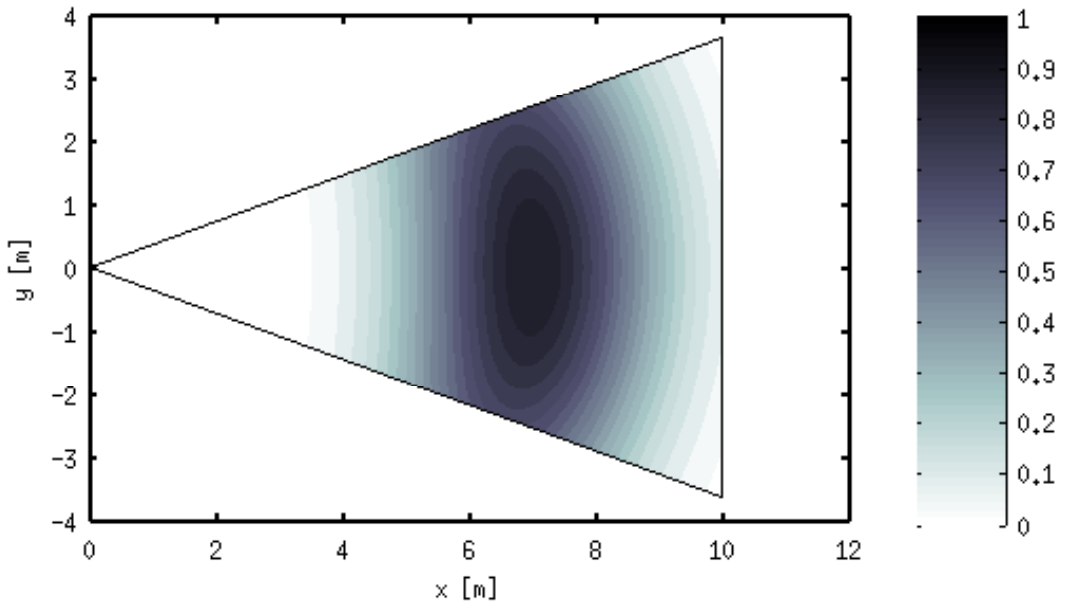


Figure 12: Experimental results showing the true and estimated object positions as measured in the body frame of the robot. The angular bias appears to be independent of the true position while the distance error is smallest for the objects placed at $x = 8$ m. Performance significantly degrades at the $x = 12$ m line. This is overlaid on the binary detection model, where darker shading indicates a higher probability of a measurement.

error using (3.16) and (3.17).

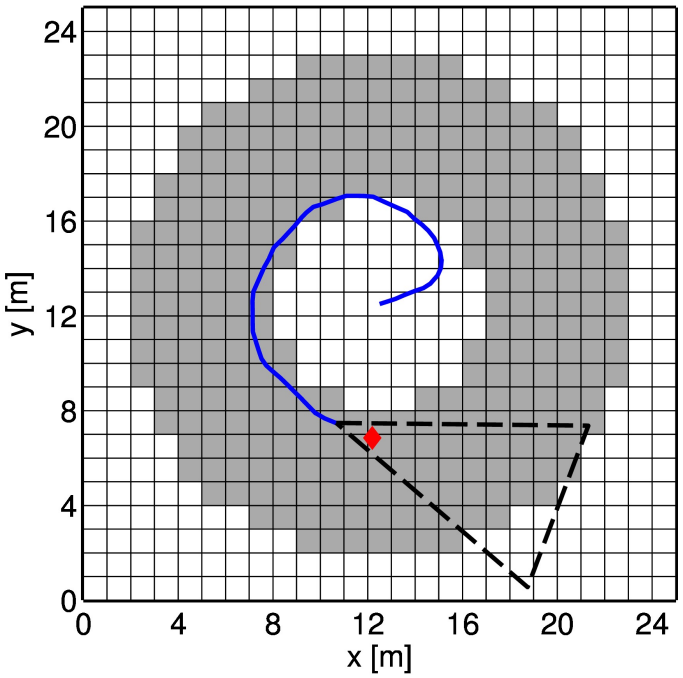
In the event that the estimate has nearly converged within the footprint of the sensor, the mutual information and its gradient will be near zero so the local, greedy controller may get stuck. Longer time-horizon path planning would be the best way to prevent this, however even with the reductions in complexity, mutual information is prohibitively expensive for such searches. Instead, when mutual information is below some threshold, $\tau_I \ll 1$, the robot drives toward the cell with the highest entropy in the probability of occupancy, *i.e.*, with probability nearest 0.5. The intuition here is that, because maximizing the expected reduction in entropy due to a sensor reading is equivalent to maximizing mutual information, driving toward the cell with the highest uncertainty will still lead to the desired behavior. Note that this choice ignores uncertainty in sensing and only considers marginal distributions of $p(X)$ over individual cells. While this is sufficient to perturb the robot away from local extrema in the greedy controller, it will not perform as well for local searches.

3.5.3 Test Results

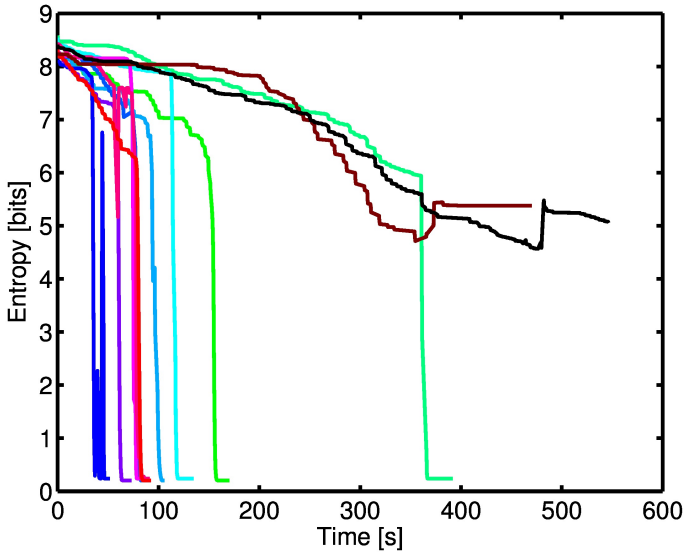
The environment used for field tests with the robot, shown in Figure 13a, is the simplest example of a non-trivial topology in the prior belief. In this scenario the robot begins at the center of the environment and searches for targets in an annular region surrounding it, where the shaded cells have non-zero probability of occupancy in the prior. We tested two separate cases: one where there is a single object in the environment and the robot believes that there is either zero or one targets within the environment, and the second case where there are two targets (red dog bone toys) and the robot believes there are up to four. In the second case there were also multiple false targets (dog toys of varying color and shape) placed within the environment.

Single Object

We performed twelve trials with random initial positioning of the object. The robot knows that there is a maximum of one target within the environment. Since the total number of random finite sets is small (the number of cells plus the empty set) the cell size is fixed at 1 m. Figure 13b shows the time history of the entropy of the target distribution. In ten runs the robot correctly located the object within the precision of the grid. Initially the entropy decreases slowly as the robot sweeps out some of the area. The sudden drop is due to the fact that the number of objects is limited to one: this causes the distribution to rapidly converge when multiple detections are made in the same cell, as that means that the object cannot be in any of the other cells. The variation in time to convergence is due to the random placement of the object, with short times corresponding to the object being placed nearer the initial footprint of the robot. The robot failed to localize the object after a full sweep of the environment in two runs due to failures in the perception system, likely due to adverse lighting conditions. The system was able to recover in one such instance (dark blue line in Figure 13b), nearly converging to the incorrect cell before switching to the correct cell, causing the large spike in entropy near the end of the trial.



(a) Prior belief and path



(b) Entropy

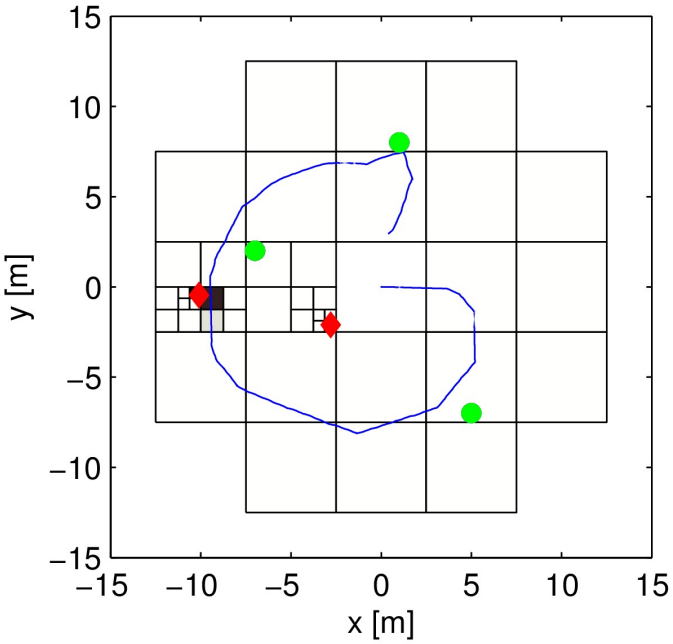
Figure 13: Sample results from experimental data. (a) A typical path taken by the robot, starting from the center of the annulus, is indicated by the solid line. The final position of the robot and its sensor footprint are given by the dashed line. The true object location is given by the red diamond and clutter objects by green circles. Shaded cells correspond to non-zero prior probability of the cell containing an object. (b) Time history of the entropy of the distribution of object locations over 12 representative runs.

Two Objects

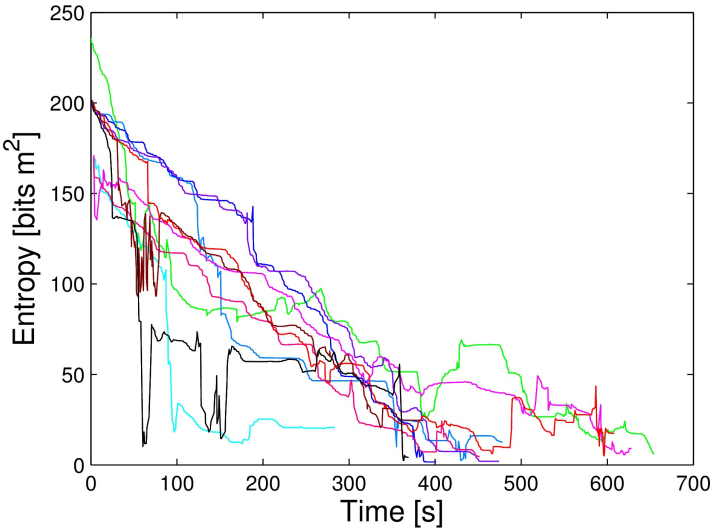
In this case, the robot begins with a coarse grid of 5 m cells with a minimum resolution of 0.625 m. These trials take noticeably longer to complete as the robot must sweep out the entire area to determine that there are no targets present in cells that are unexplored. This is a result of the maximum number of possible targets (four) being greater than the true number of targets (two). It also means that the entropy will not drop as sharply when a target is detected. Figure 14b shows the time evolution of the entropy, where the units are given in bits·m^2 to take the variable cell size into account since a large cell has higher uncertainty in the location of the target compared to a small cell with the same probability of occupancy. Initial true and clutter object locations were random, in some cases with true objects within 1 m of one another. In the ten trials the robot only failed to detect one of the twenty total targets, with the failure due to a sudden change in the lighting conditions outdoors. Figure 14a shows the results of a typical run, where the left-most target was not perfectly localized (the cell to the right has non-zero probability of occupancy) due to the object being located right on the cell boundary. However, over the course of these experimental runs (1–10 mins) the robot experiences insignificant drift in the position estimate. This may become an issue for much larger environments where the robot would be in use for longer periods of time. The robot also never localized a clutter object despite several isolated false positive detections.

3.6 Quadrotor Experiments

We have also tested this framework using a small team of Ascending Technology Hummingbird quadrotor MAV robots equipped with magnetometer sensors. Magnetometers measure the strength of the local magnetic and, in this situation, are used to detect anomalies in order to localize targets. This is known as Magnetic Anomaly Detection (MAD) and is used in geological surveys to detect ore deposits, in military reconnaissance to detect submerged submarines, and other such tasks. Such MAD sensors are finely calibrated to detect very subtle disturbances in the Earth's magnetic field. The noisy, scaled-down laboratory



(a) Prior belief and path



(b) Entropy

Figure 14: Sample results from experimental data. (a) A typical path taken by the robot, starting from the center of the annulus, is indicated by the solid line. The true object locations are given by the red diamonds. Shaded cells correspond to non-zero prior probability of the cell containing an object. (b) Time history of the entropy of the distribution of object locations over ten representative runs.

environment is representative of more complex, real-world environments.

These sensors are much lower cost and, as a result, are much coarser. Additionally, it is well known that magnetometers are not reliable indoors due to the presence of metal building materials, electrical wiring, and other such components. This problem is further exacerbated by the robots themselves: the drive motors contain permanent magnets, the wires to the drive motors have very high current and the magnetometers are located near the onboard computers and wireless antennae. Furthermore, the strength and direction of a magnetic field depends highly on the orientation of the magnetic source. This makes the inference problem difficult.

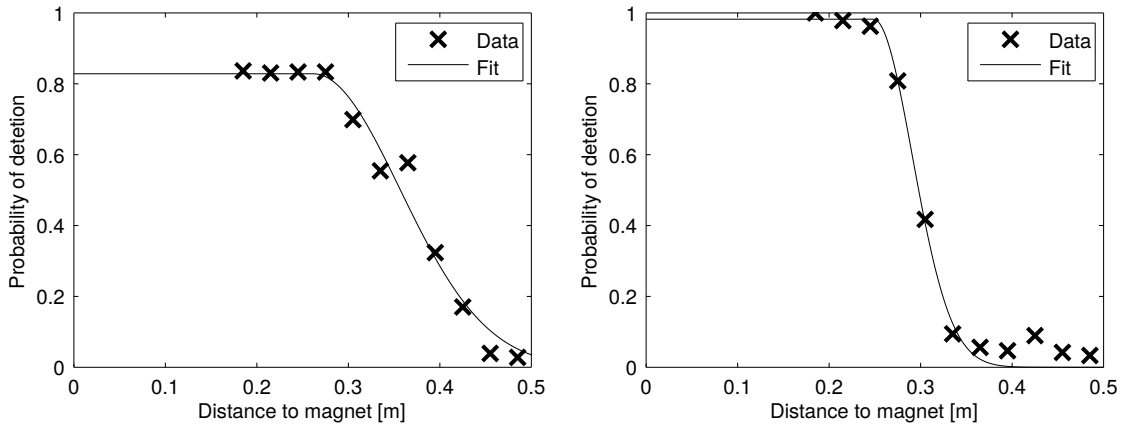
To account for all of these uncertainties we model the magnetometer as a binary sensor, returning a positive measurement if the magnetic field is "sufficiently" disturbed from the nominal value. The probability of detection takes the form

$$p_d(\mathbf{x} \mid \mathbf{q}) = \begin{cases} 1 - p_{\text{fn}} & |\mathbf{x} - \mathbf{q}| < R_0, \\ (1 - p_{\text{fn}}) \exp\left(-\frac{(|\mathbf{x}-\mathbf{q}|-R_0)^2}{2\sigma_R^2}\right) & R_0 \leq |\mathbf{x} - \mathbf{q}| \leq R_{\max}, \\ 0 & R_{\max} < |\mathbf{x} - \mathbf{q}|. \end{cases} \tag{3.31}$$

The values of the parameters $p_{\text{fn}}, \sigma_R, R_0$ depend on the specific sensor and robot being used. Figure 15 shows the experimentally derived sensor models for the two quadrotor platforms we use, Kilo and Papa. Despite a nearly identical setup the detection models were quite different, likely due to small differences in the locations of onboard computers and wireless transmitters. See Appendix A for further details on the sensor characterization.

3.6.1 Single Robot Results

We conduct a series of hardware and simulation experiments with a single robot to test the performance of the search algorithm in the MAD setting and to validate the performance of the simulation environment. Each robot performs three individual hardware trials and five simulation trials. The hardware experiments are performed in a Vicon motion capture system, which provides each robot with an accurate estimate of its pose. The robots explore

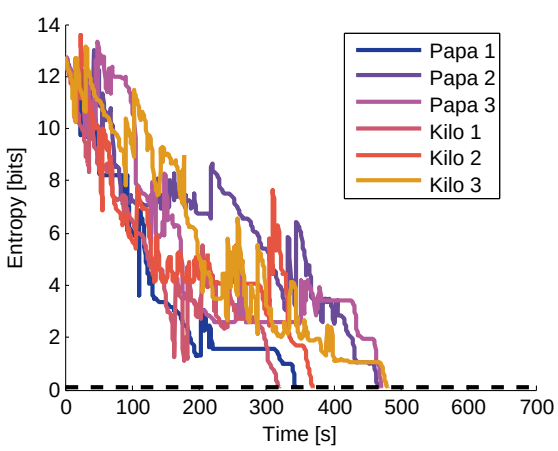


(a) Quadrotor Kilo (b) Quadrotor Papa

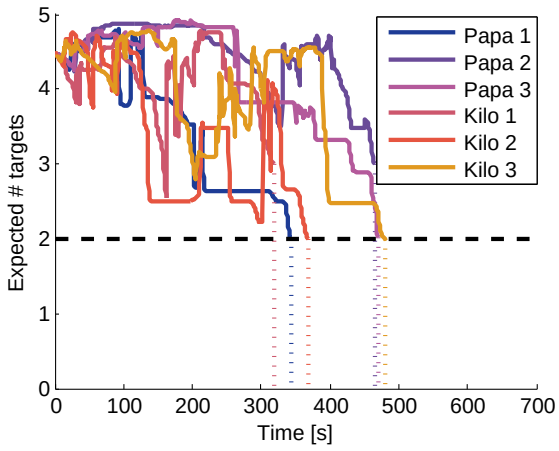
Figure 15: Experimentally determined MAD sensor detection models used for target detection and localization.

a 2×2 m area that is divided into cells with a maximum edge length of 50 cm and a minimum length of 12.5 cm. The minimum cell size is on the same length scale as the 10 cm long targets and significantly smaller than the 1 m diameter sensor footprint.

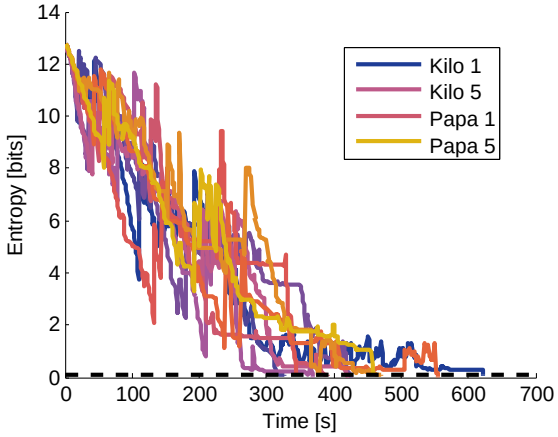
Figure 16 shows the resulting evolution of the entropy of target set and of the expected number of targets over time. The overall behavior is similar across both hardware and simulation experiments. The target entropy decreases quickly at first and has several step increase as cells are subdivided before finally reaching the desired level of 0.1 bits. Similarly, the expected number of targets begins near 4.5 before reaching a final value near 2.0, the true value. For both the simulated and hardware experiments, there was a single trial where the final expected number of targets was 3.0. Figure 17b shows the final estimate for one such occurrence, where the robot incorrectly determined that two adjacent cells both contain a target. There was also a single trial in both the hardware and simulated systems where one of the targets was mis-localized, with the true target being in a cell adjacent to the final estimated position. Figure 18 shows the statistics of the time to completion. The minimum and median times are very similar, at 320 s and 417.5 s for the hardware experiments and 326 s and 422.5 s for the simulation experiments.



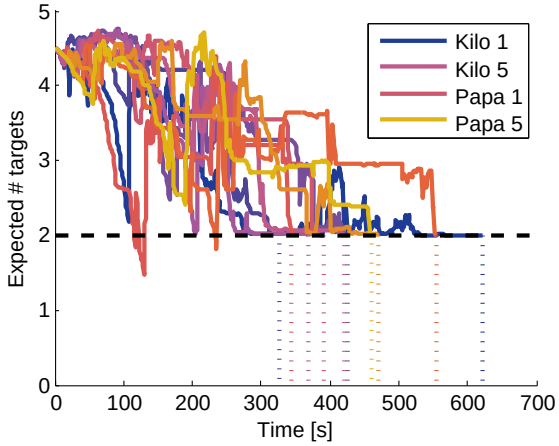
(a) Hardware experiments – entropy



(b) Hardware experiments – expected number of targets

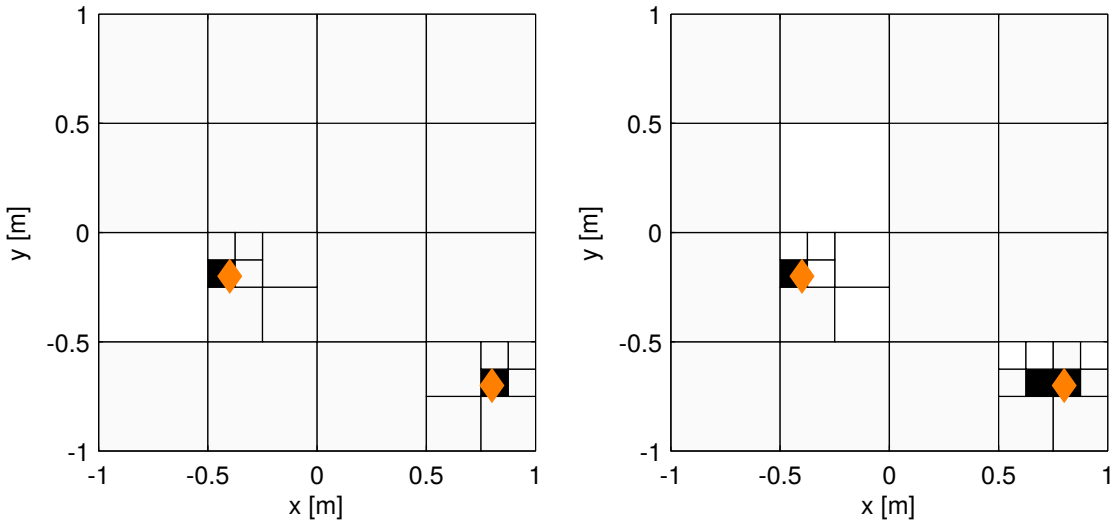


(c) Simulation experiments – entropy



(d) Simulation experiments – expected number of targets

Figure 16: Experimental results for single robot experiments. Hardware results are shown in (a) and (b) and simulation results in (c) and (d). The time evolution of the target entropy is shown in (a) and (c) and the time evolution of the expected number of targets in (b) and (d).



(a) Hardware – Kilo

(b) Hardware – Papa

Figure 17: Localization results for a single real-world quadrotor. The orange diamonds indicate the true target positions and shading within each cell is the probability of occupancy.

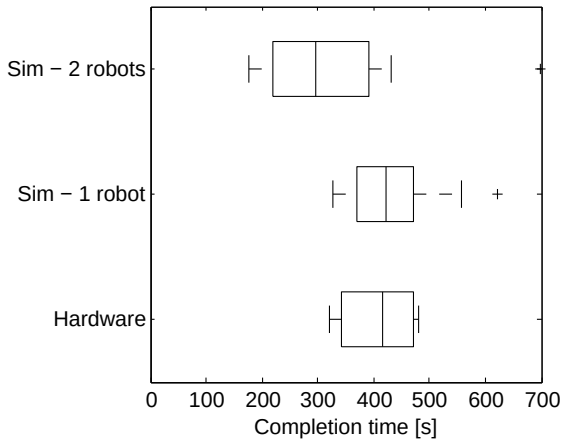


Figure 18: Box plots of the time to completion for the simulated and hardware MAD experiments.

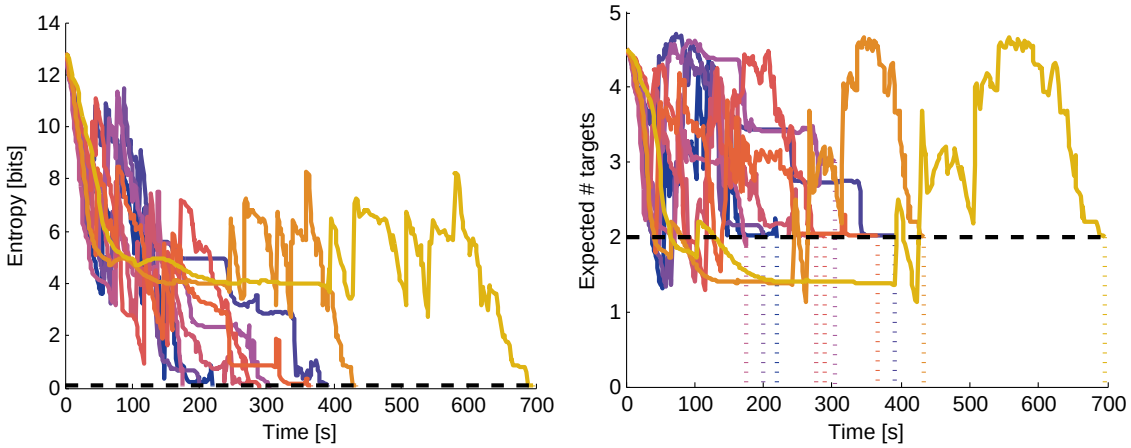
3.6.2 Two Robot Results

We use the simulation environment to test the performance with a team of two MAVs, avoiding complex, unmodeled interactions between physical robots, such as the magnetic field induced by the motors of one robot interfering with the magnetometer readings on the other robots. Given the level of similarity between the performance of the MAD system in the previous hardware and simulation experiments, we feel confident that the simulation results could be replicated in hardware. The sensor models used in these trials match those of the previous experiments, with one MAV matching Kilo and the other matching Papa so that the team had heterogeneous sensor models. Figure 19 shows the results of the simulated experiments and Figure 18 shows the statistics of the time to completion. Note that the entropy initially decreases more quickly compared to the single robot trials and that, in general, the team is able to complete the task more quickly.

There is one outlier in the two robot experiments, the gold line in Figure 19. This outlier in the time to completion is due to the team reaching a temporary deadlock due to the collision avoidance algorithm used in the experiments. The collision avoidance algorithm is myopic, with a robot backing up, *i.e.*, following the negative gradient, if it would come into collision with another robot. This occasionally causes the robots to step forwards then backwards repeatedly while the target is located near the edge of the sensor footprint, making target localization more difficult and unreliable since the detection likelihood is lowest at the edge of the footprint. This behavior also occurred to a lesser extent in one other trial, indicated by the orange line in Figure 19.

3.7 Conclusion

This chapter proposes an approximate, decentralized multi-robot control policy based on finite set statistics that allows for significant reductions in the computational complexity with small error. Recursive Bayesian filters maintain the robots' beliefs about the targets and hazards within the environment while the robots follow the gradient of mutual information, locally maximizing the expected information gain at each time step. When computing their



(a) Two robot simulations – entropy

(b) Two robot simulations – expected number of targets

Figure 19: Simulation results for two robot experiments. (a) The time evolution of the target entropy. (b) The time evolution of the expected number of targets.

control actions, robots only consider other robots with overlapping sensor footprints. This is based on the fact that real sensors and hazards have a limited range of influence in the environment, and leads to significant computational savings. Simulations illustrate that the control error due to this approximate decoupling is small in most cases.

The proposed estimation, control, and communication framework is validated through a series of simulated and hardware experiments. The first set of experiments utilizes a large ground robot equipped with a monocular camera. This robot explores an open field environment in search of a small number of targets distributed among a larger number of clutter objects. Despite this, the robot is able to reliably find the true object locations. The second set of experiments utilizes a team of one to two quadrotor MAV platforms equipped with magnetic anomaly detection sensors. These robots explore an open environment in search of magnets, reliably determining the true number of targets and their positions. We validate the simulation environment by comparing the results of the simulated and hardware experiments. This simulation environment is then used to demonstrate the performance of a team of robots with heterogeneous detection statistics. The team is able to effectively coordinate to search for multiple targets.

Chapter 4

Active Detection and Localization of a Large Number of Targets

We are interested in applications such as search and rescue, security and surveillance, and smart buildings and smart cities, in which teams of mobile robots can be used to explore an environment to search for a large, unknown number of objects of interest. Concrete examples include using thermal imaging to locate individuals trapped in a building after a natural disaster, using cameras to locate suspicious packages in a shopping center, or using wireless pings to locate sensors within a smart building or smart city. Real-world examples of such smart building scenarios include Rowe et al. [88], which features thermostats, microphones, access points, and bluetooth-enabled actuators within a building, and Fu [35], which describes low-power sensors embedded within construction materials. In each of these examples, the number of objects is not known a priori, and can potentially be very large. The sensors used by the robots to localize targets can be noisy, there can be many false positive or false negative detections, and it may not be possible to uniquely identify and label individual objects.

This chapter uses the Probability Hypothesis Density (PHD) filter, described in Chapter 2, to detect and localize an unknown number of targets. The robots jointly plan actions that maximize the mutual information between the resulting multi-target estimate and the

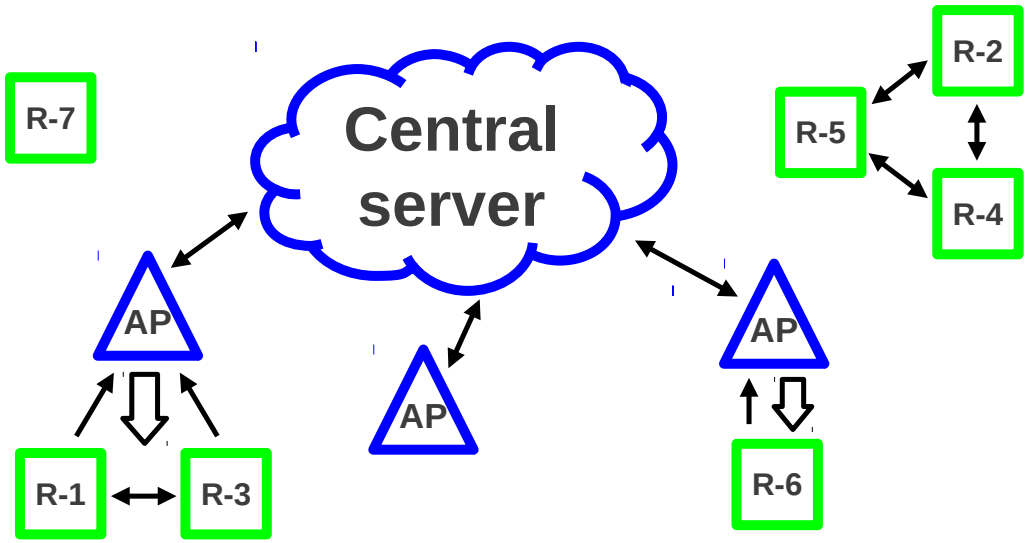


Figure 20: Diagram of the decentralized network structure. Robots (green squares) are able to communicate on a peer-to-peer basis with nearby robots as well as access the central server through access points (blue triangles). The communication links originating from robots are all relatively low-bandwidth while the downlink from the server may be higher bandwidth.

future binary events indicating whether a sensor detects any targets or not, effectively hedging against uninformative actions in a computationally tractable manner. This approach offers several key advantages: scalability in the number of targets, avoidance of any explicit data association, and the ability to handle a variable number of measurements at each time step.

This chapter presents an information-based, receding horizon control law that allows small teams of robots to perform autonomous information gathering tasks. We demonstrate the real-world applicability of the proposed control law through a series of hardware experiments using a team of ground robots equipped with bearing-only sensors seeking tens of targets in an indoor office environment. We further demonstrate that the proposed control law performs well across variable team size, different environments, target cardinalities that span orders of magnitude (1 to 100), and different sensor modalities. In all of these cases, the robot team is able to accurately estimate the number of targets and the locations of the targets within the environment, autonomously concluding the exploration when the team reaches a desired level of confidence in the target localization.

Many of these tasks, such as surveillance, security, and monitoring, all take place in locations with an existing communication infrastructure that the team can leverage. The need for a communication architecture is central to the performance of a cooperative robotic team, yet must take into account the limited capabilities (*e.g.*, communication range and bandwidth) of each robot while allowing robots to exchange information in a consistent way. A centralized approach will not work over large scale environments where not all robots will be able to communicate with one another. One common decentralized architecture is Decentralized Data Fusion (DDF), first described by Grime and Durrant-Whyte [40], in which each robot manages its own copy of the joint belief and aggregates data from the other robots through channel filters which only admit information that is distinct from their current belief. The DDF framework is particularly amenable to Gaussian beliefs as the information form of the Kalman filter allows for efficient, low-bandwidth updates. However, more complicated belief representations often require overly conservative approaches to data fusion.

Our solution takes the best of fully centralized and fully decentralized network solutions, allowing robots to communicate on a peer-to-peer basis in a decentralized fashion while also including communication with a centralized server, which robots may access via the existing network infrastructure in the environment. This central server offer robots access to a centralized repository of data, to additional computational resources, and to cloud services. This idea of robots relying on information from a server has been called cloud robotics and has recently generated quite a bit of excitement [43, 44]. A similar idea was also used for estimation and control of groups of robots by Michael et al. [74] where an Asymmetric Broadcast Control (ABC) was used to synthesize locally derived information and provide low-resolution global information to the group. The asymmetry is in the communication between the robots and the server. Uploads from robots are low-bandwidth by nature but downloads involving global information may require higher bandwidth. Robots are not required to constantly communicate with the central server or cloud, instead they opportunistically upload and retrieve information based on their physical proximity to ac-

cess points. Figure 20 shows such a network architecture, where robots may have one or more communication links and can trade off the benefits of accessing the server compared to taking further local measurements.

We present a decentralized control architecture founded upon the ideas of information gathering, synthesis, and dissemination. Gathering is done using a team of mobile sensors, the only strong assumption being that robots are able to localize themselves and navigate without noise. The data is then incorporated into the robot's belief through the PHD filter, making no additional assumptions on the targets' spatial or cardinality distributions. The synthesis of peer-to-peer and cloud information is done in a principled way, synchronizing the beliefs of robots and ensuring no data is double counted as it is exchanged. Mutual information balances the benefits of obtaining information by direct observation of the environment or by downloading from the server, merging the objectives of gathering and disseminating information into a single control law.

The research in this chapter was originally published in [22, 24, 25].

4.1 Problem Formulation

This chapter considers the problem of a team of R robots exploring an environment E in search of targets. The robots are assumed to be able to localize themselves within the environment, or at least with sufficiently high accuracy so that any errors will have a negligible effect on the target localization. At time t the robot r has pose $\mathbf{q}_t^r$ and receives a set of measurements $Z_t^r = \{\mathbf{z}_{1,t}^r, \ldots, \mathbf{z}_{m_t^r,t}^r\}$, which has m_t^r measurements. A set of n target locations is given by $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, where each $\mathbf{x}_i \in E$. Here Z and X are realizations of random finite sets (RFSs), where an RFS is a set containing a random number of random elements, *e.g.*, each of the n elements $\mathbf{x}_i$ in the set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ is a vector indicating the position of a single target.

4.1.1 Sensor Models

Each robot is equipped with a sensor that is able to detect targets within its field of view (FoV). The probability of a sensor with pose $\mathbf{q}$ detecting a target at $\mathbf{x}$ is given by $p_d(\mathbf{x} \mid \mathbf{q})$

and is identically zero outside of the FoV. Note the dependence on the sensor's position, denoted by the argument $\mathbf{q}$. If a target at $\mathbf{x}$ is detected by a robot at $\mathbf{q}$ then it returns a measurement $\mathbf{z} \sim g(\mathbf{z} \mid \mathbf{x}, \mathbf{q})$. The false positive, or clutter, model consists of a PHD $c(\mathbf{z} \mid \mathbf{q})$ describing the likelihood of clutter measurements in measurement space and the expected clutter cardinality. Note that in general the clutter may depend on environmental factors.

4.1.2 Communication

As we did in Chapter 3, we would like a decentralized version of this exploration strategy. In this case, we will also consider the scenario where there is a cloud-based resource management system that robots may interact with to upload measurement data and download data collected by other robots. This cloud-based management system may have multiple access points within the environment, or robots may be able to intermittently access the resources through specialized communication channels.

As robots explore the environment, they store a local history of messages, where messages consist of (position, measurement set) pairs. This history will be shared with other robots, directly over peer-to-peer links and indirectly through the central server, to aid in exploration. The central server has A stationary access points located in the environment at $\mathbf{s}_1, \ldots \mathbf{s}_A$, at which robots upload messages and download the latest PHD from the server, $v_s(\cdot)$.

Robot-server communication, as previously noted by Michael et al. [74], is asymmetric in the bandwidth. When a robot is within communication range of an access point, the robot uploads its message history since the last check-in, waits while the server uses these messages to update its PHD v_s, and receives the resulting PHD from the server. This PHD v_s, which includes all of the robot's own message history as well as all information uploaded by other robots prior to the current time, replaces the robot's local PHD.

On the other hand, robot-robot communication is symmetric. Here robots form coalitions, which are connected components of a communication graph with edges between robots that are able to communicate. Robots then simply exchange their most recent messages with all other robots in the coalition. These messages are then used to update the PHD. This

framework allows robots to jointly explore the environment while not double-counting any information, as communication with the central server overwrites the peer-to-peer updates.

4.2 Information-Based Receding Horizon Control

In Section 2.2.3 we saw that the general Bayes filter was intractable due to the number of possible data associations, necessitating the development of the PHD filter. The same sum over data associations also appears in the expression for the mutual information between the target and measurement sets, making it prohibitively expensive to compute. To get around this, we consider the binary event of receiving an empty measurement set, as was done in Section 3.5,

$$y = \begin{cases} 0 & Z = \varnothing \\ 1 & \text{else.} \end{cases} \tag{4.1}$$

Here $y = 0$ is the event that the robot receives no measurements to any (true or clutter) objects while $y = 1$ is the complement of this, *i.e.*, the robot receives at least one measurement. Mahler proposes a similar idea [68], where the objective is to maximize the mutual information between the target set and the empty measurement set, *i.e.*, when $p(Z = \varnothing) = 1$, so

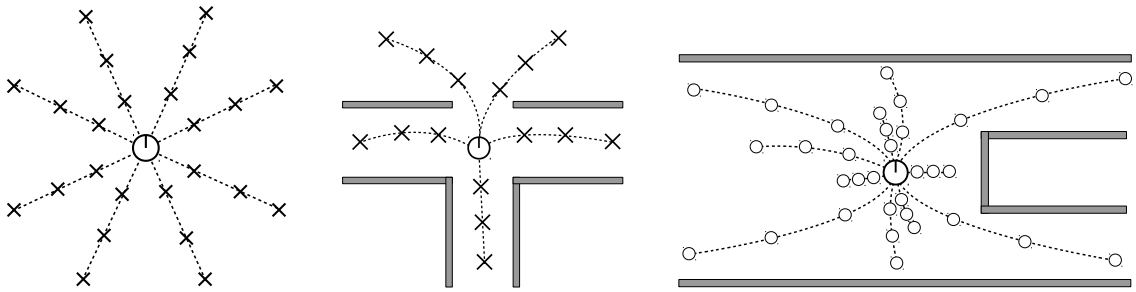
$$\mathbf{q}^* = \underset{\mathbf{q}}{\operatorname{argmax}}\, I[\mathcal{X}, \mathcal{Z}(q) = \varnothing]. \tag{4.2}$$

This objective is chosen because it hedges against the highly non-informative empty measurement set.

This paper considers the information gathering problem in a receding horizon framework, planning T actions into the future. Let the time horizon be $\tau = \{t+1, \ldots, t+T\}$. The information-based objective is then

$$\mathbf{q}_\tau^* = \underset{\mathbf{q}_\tau \in Q_\tau^{1:R}}{\operatorname{argmax}}\, I[\mathcal{X}_{t+T}; \mathcal{Y}_\tau^{1:R} \mid \mathbf{q}_\tau], \tag{4.3}$$

where $\mathcal{X}_{t+T}$ is the predicted location of the targets at time $t+T$ and $\mathcal{Y}_\tau^{1:R}$ is the collection of binary measurements for robots 1 to R from time steps $t+1$ to $t+T$, which depend on



(a) Free space (b) Cluttered environment (c) Multiple length scales

Figure 21: Example action sets for a robot in (a) free and (b) cluttered space over a horizon of $T = 3$ steps. Each action is a sequence of T poses at which the robot will take a measurement and there are 8 and 5 actions in the two sets, respectively. (c) Shows an example action set over multiple length scales.

the future locations of the robots $\mathbf{q}_\tau = [(\mathbf{q}^1_{t+1})^T, \ldots, (\mathbf{q}^1_{t+T})^T, \ldots, (\mathbf{q}^R_{t+T})^T]^T$. Note that the robot poses $\mathbf{q}$ are *not* random variables themselves, but the random variable $\mathcal{Y}^r_t$ depends on the value $\mathbf{q}^r_t$ through the detection model $p_d(\mathbf{x} \mid \mathbf{q})$. Future work will take into account the uncertainty in the poses of the robots.

4.2.1 Action Set Generation

The possible future measurements of the robots depend upon their future locations within the environment, so the action set for the team, $Q^{1:R}_\tau$, must be sufficiently rich for the robots to explore the environment. Simultaneously, it must be kept as small as possible to reduce the computational complexity of (4.5).

Over a short time horizon, an individual robot may move in a small neighborhood around its current location. If one were to naïvely chain such actions, there would be a exponentially growing number of possible actions. However, many of these would be redundant, *i.e.*, the robots would traverse the same region. Atanasov et al. [3] present one solution to this problem, pruning the tree of motion primitives to eliminate uninformative and redundant actions. We take an alternative approach to curb the number of actions while maintaining diversity. Each robot selects a number of candidate points at a given length scale from its current location, plans paths to those goals, and interpolates the paths to get T intermediate points, as shown in Figure 21. This forms the basis of actions Q^r for an individual robot r at a particular length scale.

Algorithm 5 Action Set Generation

procedure ACTIONSET($\ell, T, \mathbf{q}, M$) ▷ Action set at length scale ℓ with a horizon T for a robot at $\mathbf{q}$ in map M
  $P \leftarrow \{\mathbf{x} \in M \mid d(\mathbf{x}, \mathbf{q}) = \ell\}$ ▷ All points in the map a distance ℓ from robot
  $G \leftarrow \{\mathbf{x}\}$ ▷ Pick any point $\mathbf{x} \in P$
  $Q \leftarrow \varnothing$ ▷ Action set
  while P is NOT empty **do**
    $\mathbf{x}^* = \text{argmin}_{\mathbf{x} \in P} \min_{\mathbf{y} \in G} d(\mathbf{x}, \mathbf{y})$ ▷ Find point nearest to existing goals
    $P \leftarrow P \setminus \{\mathbf{x} \in P \mid d(\mathbf{x}, \mathbf{x}^*) < R\}$ ▷ Remove all points near the goal location
    $G \leftarrow G \cup \{\mathbf{x}^*\}$ ▷ Add to list of goals
    Path $\leftarrow$ path from $\mathbf{q}$ to $\mathbf{x}^*$ ▷ Found using A*
    $Q \leftarrow Q \cup \{\{T$ evenly spaced points along Path$\}\}$
  end while
  return Q
end procedure

It is advantageous to plan over multiple length scales, as there are some instances where a lot of information may be gained by staying in a small neighborhood, while other times it is more beneficial to travel to a distant, unexplored area. To allow for this diversity, each robot generates actions, such as those in Figure 21, over a range of L length scales. The number of planning steps, T, at each length scale is kept constant so that meaningful comparisons between the information values can be made.

Concurrent

Ideally, the team would plan over all possible actions for all robots at the same time. The individual robot action sets consist of all actions over all length scales, and the joint action set is the Cartesian product of the individual action sets, $Q^{1:R} = Q^1 \times \ldots \times Q^R$. This leads to individual robot action sets that grow linearly in the number of length scales, $|Q^r| = \mathcal{O}(L|Q_\ell|)$ and a joint action set that grows exponentially in the number of robots, $|Q^{1:R}| = \mathcal{O}((L|Q_\ell|)^R)$, where $|Q_\ell|$ is the number of actions at an individual length scale. This makes the concurrent computations prohibitively expensive for all but small teams of robots with small action sets.

Sequential

To alleviate some of the computational load, we may apply a sequential, but approximate, approach to select the best action for the team. Robots sequentially optimize over length scales, individual robots, or both. This reduces the size of a joint action set to $\mathcal{O}(|Q_\ell|^R)$, $\mathcal{O}(L|Q_\ell|)$, or $\mathcal{O}(|Q_\ell|)$, respectively, and there are L, $\mathcal{O}(R)$, or $\mathcal{O}(LR)$ action sets. However, there is no guarantee that the resulting joint action computed using any of the sequential methods is identical to the fully concurrent plan.

Length Scales To sequentially plan over length scales, the objective changes to

$$\mathbf{q}_\tau^* = \underset{\ell}{\operatorname{argmax}} \underset{\mathbf{q}_{\tau,\ell} \in Q_{\tau,\ell}^{1:R}}{\operatorname{argmax}} I[\mathcal{X}_{t+T}; \mathcal{Y}_\tau^{1:R} \mid \mathbf{q}_{\tau,\ell}], \tag{4.4}$$

where $Q_{\tau,\ell}^{1:R}$ is the set of joint actions at length scale ℓ.

Robots Planning over individual robots is slightly more complicated. The first robot plans its action independently of all other agents, and each subsequent agent plans its action conditioned on all of the other robots' paths. This cycle is repeated until the robots reach a consensus, *i.e.*, robots have a chance to update their original plans given the new plans of other agents. The sequence of joint action sets, $Q^{1:R}$, is

$$\begin{gathered}
Q^1 \times \varnothing \times \varnothing \times \varnothing \times \ldots \times \varnothing \\
\mathbf{q}^{*1} \times Q^2 \times \varnothing \times \varnothing \times \ldots \times \varnothing \\
\mathbf{q}^{*1} \times \mathbf{q}^{*2} \times Q^3 \times \varnothing \times \ldots \times \varnothing \\
\vdots \\
\mathbf{q}^{*1} \times \mathbf{q}^{*2} \times \ldots \times \mathbf{q}^{*R-1} \times Q^R \\
Q^1 \times \mathbf{q}^{*2} \times \ldots \times \mathbf{q}^{*R-1} \times \mathbf{q}^{*R} \\
\mathbf{q}^{*1} \times Q^2 \times \mathbf{q}^{*3} \times \ldots \times \mathbf{q}^{*R} \\
\vdots
\end{gathered}$$

where $\mathbf{q}^{*r}$ is the element of Q^r with the highest expected information gain given the paths of all other robots at the time of planning. This is similar to the idea of Adaptive Sequential Information Planning from Charrow et al. [12].

This sequential optimization over robots is like the idea of coordinate descent in optimization, where a utility function is optimized over each coordinate in sequence reaching until a local optimum. In practice, the number of cycles through the team until reaching consensus was typically one, and never more than three. Atanasov et al. [5] show that this "coordinate descent" will result in a 2-approximation of the objective, meaning that the value of the objective using the approximate approach will be at least half of the value of the optimal solution. Note that this holds for arbitrary ordering of the robots and does not require re-optimizing until convergence. This re-optimization step is solely to improve the empirical performance.

Both To sequentially plan over both robots and length scales, we use the objective (4.4), where the inner argmax is over the sequence of action sets from above.

Planning Modes

We use the following shorthand to describe the different planning modalities:

- Mode 0: MI, concurrent robots, concurrent length scales
- Mode 1: MI, concurrent robots, sequential length scales
- Mode 2: MI, sequential robots, concurrent length scales
- Mode 3: MI, sequential robots, sequential length scales
- Mode 4: Random, concurrent length scales
- Mode 5: Random, sequential length scales

In modes 0–3, the action is selected by maximizing the mutual information, as described in this section, while in modes 4 and 5 an action is selected randomly from the joint action set. For modes 1, 3, and 5, the length scale with the highest expected information gain is selected.

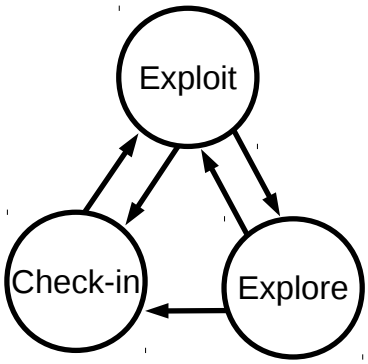


Figure 22: Finite state machine of the three control modes.

4.2.2 Finite State Machine

There are three possible motion modalities for the robots, the choice of which depends upon the recent history of the robot actions: Explore, Check-in, and Exploit. A finite state machine, Figure 22, shows the possible mode transitions. For both the Explore and Check-in modes, robots select a goal location $\mathbf{q}_g$ and plan a path there from the current location $\mathbf{q}_t^r$. In general these paths require many individual motions due to the limitations on speed, so that robots collect measurements along the way but do not react on them.

Explore If the longest length scale is too small, there may be some instances where the extended planning horizon of the robots is not sufficient to escape from an information minimum, *e.g.*, a robot finishes exploring a corner of the environment and all surrounding area has already been well explored. To avoid such situations, if a robot becomes stuck (*i.e.*, when it has not left a small neighborhood around its current position for a certain number of time steps T_S) then it selects an unexplored location in the environment and drives there, an example of the typical exploit/explore behavior in information gathering tasks. These exploration points are randomly selected with probability proportional to the probability of the team not having detected a target at that location, *i.e.*, $\mathbf{q} \sim p(\cdot) \propto \prod_{k=1}^{t} \prod_{r=1}^{R} p_d(\cdot \mid \mathbf{q}_k^r)$.

Check-in In order to keep the belief in the server (which may be monitored by a human operator) up-to-date and the robots' beliefs somewhat synchronized, robots are required to check-in with the server at least every T_C time steps. This behavior may be removed by

setting $T_C = \infty$. A robot may also enter this control mode if the expected information value of communicating with the server is higher than the information value of locally sensing the environment,

$$\mathbf{q}_\tau^* = \max \left\{ \underset{\mathbf{q}_\tau \in Q_\tau^{1:R}}{\operatorname{argmax}} I[\mathcal{X}_{t+T}; \mathcal{Y}_\tau^{1:R} \mid \mathbf{q}_\tau], \underset{a}{\operatorname{argmax}} I[\mathcal{X}_{t+T}; \mathcal{Y}_\tau^{a} \mid \mathbf{q}_\tau^{a}] \right\}, \tag{4.5}$$

where $\mathcal{Y}_\tau^a$ are the measurements available for the robot to download at the server from access point a.

Exploit If a robot enters the Exploit mode, it will look for nearby robots so that they may coordinate their actions and explore more quickly. To this end, we redefine a coalition to be a connected component of the control graph, where edges indicate that robots can communicate *and* their sensor footprints overlap, *i.e.*, $F_i \cap F_j \neq \varnothing \Rightarrow i, j \in C$. Each coalition then elects as its leader the robot that has most recently checked in with the server as the leader. The leader then plans the joint action of all robots in that coalition using its own PHD and (4.5), which in general differs from that of other robot's, in order to reduce redundancies robot motions.

4.2.3 Receding Horizon

As this is a receding horizon control law, the robots replan their action after executing a fraction of the current action. In this work, the team replans an action after each of the robots has completed at least one of the T actions, *i.e.*, after all robots have traversed $1/T$ of the planned path length. This allows robots acting at larger distance scales than others to visit at least one of their planned locations, even if the robots acting at shorter distance scales have completed their actions.

However, it is worth noting that if robots are acting at very different length scales, it is possible for one robot to have completed its full action (*i.e.*, reached all T waypoints) before one of the other robots has reached its first waypoint. In this case the first robot would sit idly, waiting for the second to trigger the replanning.

4.2.4 Computing the Objective Function

We utilize the factorization of mutual information from (2.53), $I[\mathcal{X}; \mathcal{Y}] = H[\mathcal{Y}] - H[\mathcal{Y} \mid \mathcal{X}]$, to compute the objective function in (4.5). For notational compactness, we remove the dependence of the sensor models on the pose of the robot.

Entropy

We begin by computing the binary measurement likelihoods, $p(y)$, first for an individual robot and then for a team of robots. The only way for the sensor to get no detections is for it to have zero clutter detections and to not detect any target, so

$$p(y = 0 \mid X) = e^{-\mu} \prod_{\mathbf{x} \in X} (1 - p_d(\mathbf{x}_i)), \tag{4.6}$$

where $\mu = \langle 1, c \rangle$ is the expected number of clutter detections, $c(\mathbf{z})$ is the clutter PHD from Section 4.1.1, and $e^{-\mu}$ is the probability of receiving no clutter detections given the assumption of Poisson clutter cardinality. Using this, we get that

$$\begin{aligned} p(y = 0) &= \int p(y = 0 \mid X) p(X) \, \delta X \\ &= p_K(0) \sum_{n=0}^{\infty} p(n) \underbrace{\frac{\langle 1 - p_d, \, v \rangle^n}{\langle 1, \, v \rangle^n}}_{= (1-\alpha)^n} \\ &= p_K(0) \, \langle (1 - \alpha)^n, \, p \rangle \qquad (4.7) \\ &= e^{-\mu - \alpha\lambda} \qquad (4.8) \end{aligned}$$

where $\lambda = \langle 1, \, v \rangle$ is the expected number of targets, α is the expected fraction of the targets detected

$$\alpha = 1 - \lambda^{-1} \langle 1 - p_d, \, v \rangle = \lambda^{-1} \langle p_d, \, v \rangle \, , \tag{4.9}$$

and $\mu = \langle 1, \, c \rangle$ is the expected number of clutter measurements so that $p_K(0) = e^{-\mu}$. Note that (4.7) is for the CPHD filter and (4.8) specializes the result to the PHD filter using the

fact that

$$\langle (1-\alpha)^n,\, p\rangle = \sum_{n=0}^{\infty} (1-\alpha)^n e^{-\lambda} \frac{\lambda^n}{n!} \cdot e^{\alpha\lambda - \alpha\lambda} = e^{-\alpha\lambda}.$$

This is easily extended to the multi-robot case. Let C_0 be the set of robots with $y^r = 0$ and C_1 the set of robots with $y^r = 1$. Then

$$\begin{aligned}
p(Y^1, \ldots, Y^R) &= \int \prod_{r \in C_0} p(Y^r = 0 \mid X) \prod_{r \in C_1} (1 - p(Y^r = 0 \mid X)) p(X)\, \delta X \\
&= \int \sum_{C \subseteq C_1} (-1)^{|C|} \prod_{r \in C_0 \cup C} p(Y^r = 0 \mid X) \\
&= \sum_{C \subseteq C_1} (-1)^{|C|} p_K(0)^{|C_0 \cup C|} \sum_{n=0}^{\infty} p(n) \underbrace{\frac{\left\langle \prod_{r \in C_0 \cup C} (1 - p_d^r),\, v \right\rangle^n}{\langle 1,\, v\rangle^n}}_{= (1 - \alpha(C_0 \cup C))^n} \\
&= \sum_{C \subseteq C_1} (-1)^{|C|} p_K(0)^{|C_0 \cup C|} \left\langle (1 - \alpha(C_0 \cup C))^n,\, p \right\rangle \qquad (4.10) \\
&= \sum_{C \subseteq C_1} (-1)^{|C|} e^{-|C_0 \cup C|\mu - \alpha(C_0 \cup C)\lambda}, \qquad (4.11)
\end{aligned}$$

where

$$\alpha(C) = 1 - \lambda^{-1} \left\langle \prod_{r \in C} (1 - p_d^r),\, v \right\rangle \qquad (4.12)$$

is the expected fraction of targets detected by at least one robot in group C. We substitute this in to the standard definition of entropy,

$$H[\mathcal{Y}] = -\left\langle p(y),\, \ln p(y) \right\rangle, \qquad (4.13)$$

where there are 2^{RT} possible binary measurement combinations for R robots and T time steps.

Conditional Entropy

The conditional entropy is simpler as measurement sets are conditionally independent of one another given the target set, *i.e.*,

$$p(\mathbf{y}_\tau^{1:R} \mid X) = \sum_{k\in\tau} \sum_{r=1}^{R} p(y_k^r \mid X). \tag{4.14}$$

Thus the conditional entropy of the joint measurements is simply the sum of the conditional entropies of the individual measurements, so we only need the single measurement equation

$$H[\mathcal{Y} \mid \mathcal{X}] = -\int \left(\sum_{y\in\{0,1\}} p(y \mid X) \ln p(y \mid X) \right) p(X)\, \delta X. \tag{4.15}$$

We separate the two cases for y, beginning with $y = 0$.

$$\begin{aligned}
&\int p(Y=0 \mid X) p(X) \ln p(Y=0 \mid X)\, \delta X \\
&= \sum_{n=0}^{\infty} \frac{1}{n!} \int p_K(0)\, n!\, p(n) \prod_{i=1}^{n} \big(1 - p_d(\mathbf{x}_i)\big) \frac{v(\mathbf{x}_i)}{\langle 1,\, v\rangle} \ln \left(p_K(0) \prod_{j=1}^{n} \big(1 - p_d(\mathbf{x}_j)\big) \right) d\mathbf{x}_1 \ldots d\mathbf{x}_n \\
&= p_K(0) \ln p_K(0) \sum_{n=0}^{\infty} p(n)(1-\alpha)^n \\
&\quad + p_K(0) \sum_{n=0}^{\infty} p(n) n (1-\alpha)^{n-1} \underbrace{\frac{\langle (1-p_d)\ln(1-p_d),\, v\rangle}{\langle 1,\, v\rangle}}_{-\beta} \\
&= p_K(0) \ln p_K(0)\, \langle (1-\alpha)^n,\, p\rangle - p_K(0)\, \beta\, \left\langle n(1-\alpha)^{n-1},\, p \right\rangle \qquad (4.16) \\
&= -(\mu + \beta\lambda) e^{-\mu - \alpha\lambda}. \qquad (4.17)
\end{aligned}$$

Here (4.16) is for the CPHD filter and (4.17) specializes the result to the case of the PHD filter. The negative sign in β is due to the entropy-like definition. Note for a Poisson RFS, $\left\langle n\alpha^{n-1},\, p\right\rangle = \lambda e^{-\alpha\lambda}$.

Next we examine the $y = 1$ case, using the Taylor series $\ln(1-x) = -\sum_{k=1}^{\infty} \frac{x^k}{k}$, where

$\{r\}^k$ is a set with k copies of the robot r.

$$
\begin{aligned}
&\int p(Y=1 \mid X)p(X)\ln p(Y=1 \mid X)\,\delta X \\
&= \int (1-p(Y=0 \mid X))p(X)\ln(1-p(Y=0 \mid X))\,\delta X \\
&\approx \int (1-p(Y=0 \mid X))p(X)\sum_{\ell=1}^{\infty}\frac{-p(Y=0 \mid X)^{\ell}}{\ell}\,\delta X \\
&= \int \left(-p(Y=0 \mid X)+\sum_{\ell=2}^{\infty}\frac{p(Y=0 \mid X)^{\ell}}{\ell(\ell-1)}\right)p(X)\,\delta X \\
&= -p_K(0)\left\langle (1-\alpha)^n,\, p\right\rangle + \sum_{\ell=2}^{\infty}\frac{p_K(0)^{\ell}}{\ell(\ell-1)}\left\langle (1-\alpha(\{r\}^{\ell}))^n,\, p\right\rangle \qquad (4.18) \\
&= -e^{-\mu-\alpha\lambda}+\sum_{\ell=2}^{\infty}\frac{1}{\ell(\ell-1)}e^{-\ell\mu-\alpha(\{r\}^{\ell})\lambda} \qquad (4.19)
\end{aligned}
$$

where we use the Taylor series approximation $\ln(1-x) \approx -\sum_{\ell=1}^{\infty}\frac{x^{\ell}}{\ell}$, $\{r\}^{\ell}$ is a set with ℓ copies of the robot r, (4.18) is for the CPHD filter, and (4.19) is for the PHD filter. Note that $\alpha = \alpha(\{r\})$ for a single robot and that $\alpha(\{r\}^k)$ may be computed using (4.12), where we use the first 10 terms in the Taylor series.

Server Information

The mutual information due to possible measurements in the server is more difficult to model, as the number of such measurements and the locations at which they were taken are unknown until the robot has reached an access point. Since robots do not know the locations at which measurements were taken, we assume that measurements are independent of one another (which is true provided that sensor footprints do not overlap). In this case, mutual information may be written as

$$I[\mathcal{X};\mathcal{Z}_s \mid \mathbf{q}_i] = \mathbf{E}[m]\, I[\mathcal{X};\mathcal{Z} \mid \mathbf{q}_i] \qquad (4.20)$$

where $\mathbf{E}[m]$ is the expected number of messages in the server and $I[\mathcal{X};\mathcal{Z} \mid \mathbf{q}_i]$ is the information for a single message.

To compute the information value for a single measurement, we average over possible

poses of the robot,

$$p_d(\mathbf{x}) = \int p_d(\mathbf{x} \mid \mathbf{q}) p(\mathbf{q})\, d\mathbf{q}. \tag{4.21}$$

To evaluate this, we assume $p(\mathbf{q})$ to be a uniform distribution and approximate it with a uniform grid of reachable poses over the environment, $\mathbf{q}_k$,

$$p_d(\mathbf{x}) \approx \frac{1}{N} \sum_{k=1}^{N} p_d(\mathbf{x} \mid \mathbf{q}_k). \tag{4.22}$$

It only remains to model the expected number of new measurements available in the server. Assuming that there is an average rate of return, $\rho \approx 1/T_C$, then a geometric distribution models the discrete waiting time between events. The number of messages in the server will be equal to $\tau^r - k$, where τ^r is the number of time steps since the robot under consideration last communicated with the server (*i.e.*, the length of the local message history) and k is the number of time steps for another robot. Finally, assuming robots move independently, since there are $N-1$ other robots we have:

$$\mathbf{E}\left[m\right] = (N-1) \sum_{k=0}^{\tau^r} (\tau^r - k)(1-\rho)^k \rho. \tag{4.23}$$

Computational Complexity

The computational complexity of the entropy computations is $\mathcal{O}(|Q^{1:R}| 2^{2RT})$, where R is the number of robots, T is the planning horizon, and $|Q^{1:R}|$ is the action set (described in further detail in Section 4.2.1).

4.2.5 Exploration Termination Criterion

It is unclear when to terminate the exploration in the multi-target localization problem, since the number of targets being sought is unknown. Ideally the robots should identify the exact number of targets and their locations within the environment, *i.e.*, if there are N targets in the environment then the PHD should be N Dirac delta functions, each of unit weight, centered at the true target locations. In reality, the estimates will never be as

precise, but the difference between the estimated PHD and its idealized counterpart may be used to detect when the robots have sufficient confidence in their estimate. Note that there is no way to determine the team's confidence in the cardinality estimate, as the covariance is equal to the mean for a Poisson distribution. Thus we assume that the team is able to accurately estimate the target cardinality, *i.e.*, $\lambda \to N$ as $t \to \infty$.

We turn our attention to the entropy of a Poisson RFS,

$$H[\mathcal{X}] = \lambda - \int v(\mathbf{x}) \log v(\mathbf{x})\, d\mathbf{x} \tag{4.24}$$

$$= \lambda + \lambda\big(H[\bar{v}(\mathbf{x})] - \log \lambda\big), \tag{4.25}$$

where $\bar{v}(\mathbf{x}) = \lambda^{-1} v(\mathbf{x})$ is a probability distribution created by normalizing the PHD. The ideal PHD $v^*(x)$ consists of λ particles of unit weight, so $\bar{v}^*(\mathbf{x})$ has λ particles of weights λ^{-1} and has an entropy of $\log \lambda$. The term in parentheses in (4.25) is the difference between the current normalized PHD and its idealized counterpart. The proposed termination criterion can thus be written in the two equivalent forms,

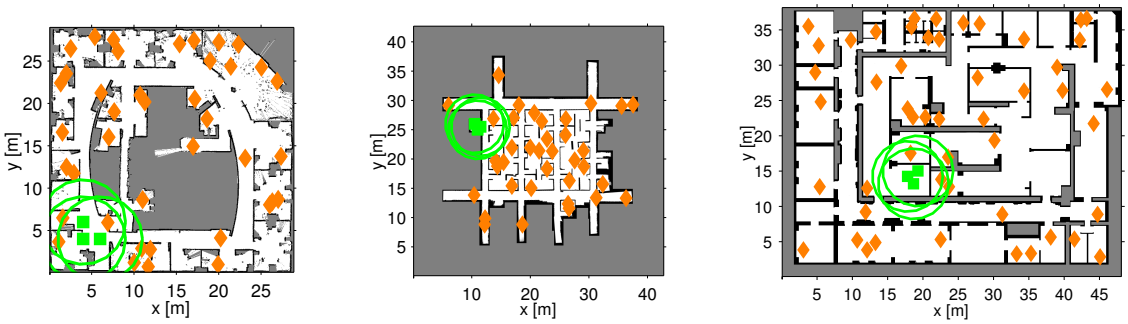
$$H[\bar{v}(\mathbf{x})] - \log \lambda \leq \epsilon \tag{4.26}$$

$$\lambda^{-1} H[\mathcal{X}] - 1 \leq \epsilon. \tag{4.27}$$

4.3 Framework Verification

To verify the performance of the proposed control strategy (4.5), simulations of three-robot teams were conducted in the three test environments shown in Figure 23. The environments contain between 36 and 50 targets for the team to localize and represent typical office buildings, where data-collecting sensors may be embedded for tasks such as climate monitoring and control. In this case the targets are stationary, so the transition model $f(\mathbf{x}, \xi)$ is the identity map, the birth PHD is identically zero, the birth cardinality is $p_\Gamma(0) = 1$, and the survival probability $p_s = 1$.

The robots are equipped with range-only sensors and are assumed to be able to travel



(a) Environment 1 (b) Environment 2 (c) Environment 3

Figure 23: Maps used in simulation runs. Robots are indicated by the green squares, sensor footprints by the green circles, and targets by the orange diamonds.

in a 2-D map, *e.g.*, differential drive robots. The detection model for the sensors, $p_d(\mathbf{x} \mid \mathbf{q})$, is shown in Figure 24 and measurements are assumed to have Gaussian noise, so $\mathbf{z} \sim \mathcal{N}(|\mathbf{x} - \mathbf{q}|, \sigma_g^2)$, where $\sigma_g = 1$ m. Clutter detections are assumed to be uniform over the sensor footprint, so $c(\mathbf{z}) = 1/5$ and the clutter cardinality $p_K(n) = \text{Poisson}(n; \mu)$, where $\mu = 0.6$. For the CPHD filter, the clutter cardinality is a truncated Poisson distribution, *i.e.*, $n \in \{0, \ldots, 8\}$ and the values are normalized so that $p_K(n)$ is a distribution.

The initial estimate of the target cardinality is 100 and the PHD is uniform over the environment. Since the targets are stationary, the PHD is represented by a uniformly-spaced grid of stationary particles. This means that the number of particles is fixed for the duration of the experiment and no resampling is required. This is equivalent to a histogram filter over a uniform grid.

To generate the trajectory basis for a single robot, we pre-compute as much as possible to allow for fast run-time performance. To do this, a uniformly-spaced set of points is laid over the environment and all-pairs shortest paths is computed using the Floyd-Warshall algorithm. To select candidate goals, the robot finds all points at a certain distance (6 m) from its current position and then prunes the set so that all goals are at least some fixed distance from each other (2 m). The time horizon is $T = 3$ steps, so the step size is 2 m. The path length considered in this work is on the same scale as the radius of the sensor's field of view, so each plan includes information about portions of the environment not currently

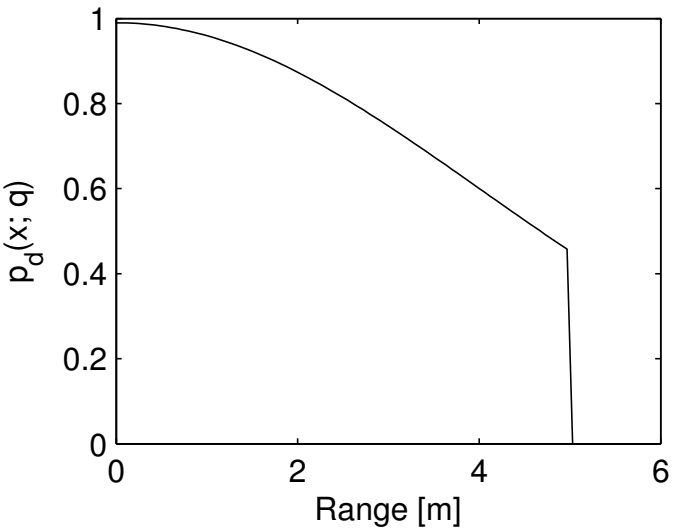


Figure 24: Detection model used in these simulation, which is independent of the bearing and because the FoV is so small it ignores environmental effects on radio signals, *e.g.*, walls, multi-path.

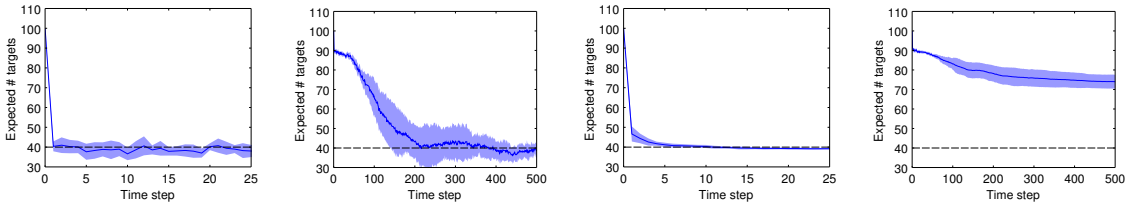
visible to the robot.

4.3.1 CPHD Filter Performance

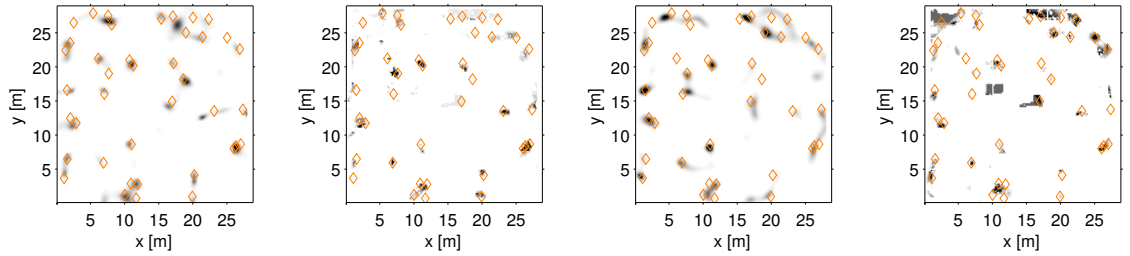
To test the performance of the CPHD filter relative to that of the PHD filter, we conduct a series of simulations in environment 1, which contains 40 targets. We run 10 trials of four configurations, PHD versus CPHD filter, and infinite versus finite FoV. In this case the clutter rate per unit of sensed area was kept constant, so the infinite FoV (set to 50 m) has a clutter rate of $\mu = 60$.

Figure 25 shows the results of these simulations, with the average expected target cardinality estimates plotted over time as well as typical PHD estimates of the target locations. As can be seen, when the FoV is infinite both the PHD and CPHD filter quickly get the true number of targets in the environment, and the low variance indicates that the estimates are consistent across trials. The CPHD filter is more consistent and the mean is smoother, supporting the claim that it handles false positive detections better than the PHD filter. Also, in individual trials with the CPHD filter, the variance on the number of targets is effectively zero ($\approx 10^{-12}$) after 15 time steps.

However, in the finite FoV case the story is much different. Here the PHD filter eventually



(a) Cardinality – PHD, 50 m FoV (b) Cardinality – PHD, 5 m FoV (c) Cardinality – CPHD, 50 m FoV (d) Cardinality – CPHD, 5 m FoV



(e) PHD – PHD, 50 m FoV (f) PHD – PHD, 5 m FoV (g) PHD – CPHD, 50 m FoV (h) PHD – CPHD, 5 m FoV

Figure 25: Expected target cardinality averaged over 10 trials for a team of robots using the PHD filter with (a) 50 m FoV and (b) 5 m FoV, and the CPHD filter with (c) 50 m FoV and (d) 5 m FoV in environment 1. The solid blue line is the average across trials and the shaded blue area is ± 1 standard deviation. Typical final PHD estimates are shown in (e)-(h) for the same configurations with the true target locations denoted by the orange diamonds.

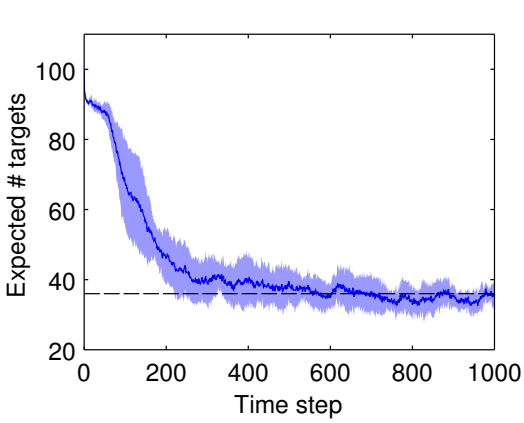
reaches the correct target cardinality, and the low variance across trials at large time steps shows that the results are consistent. On the other hand, the CPHD filter is both less accurate and more precise, never getting the correct number of targets while having low variance (≈ 1) in the target cardinality in each individual run. The same was true when starting with different initial cardinality estimates between 20 and 100. The exact causes of this phenomenon are unknown, and, to the best of our knowledge, this effect has not been previously documented in the literature.

4.3.2 Indoor Environment Simulations

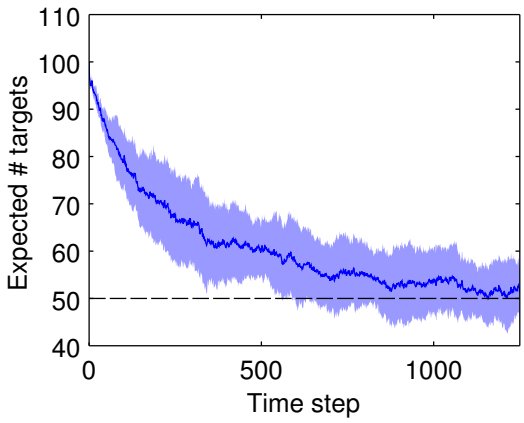
Having determined that for finite FoV sensors the PHD filter is both computationally faster and performs better at cardinality estimation, we wish to further verify its performance in other indoor environments. The results of a series of simulations in environments 2 and 3, which contain 36 and 50 targets respectively, are shown in Figure 26. Note that we again see that the average expected target cardinality asymptotes to the true number of targets and that the standard deviation across the trials tends to decrease over time, showing that the estimates are consistent.

Figure 26 also shows typical final PHDs in each environment, and we see that the targets are generally well localized. The estimation is not perfect: there are occasional false positive detections and densely packed targets are occasionally represented by a single cluster of high mass rather than individual clusters of unit mass. This is to be expected when the filter lacks target labels because a noisy measurement from one target looks equivalent to a good measurement of a nearby target, in the absence of target labels.

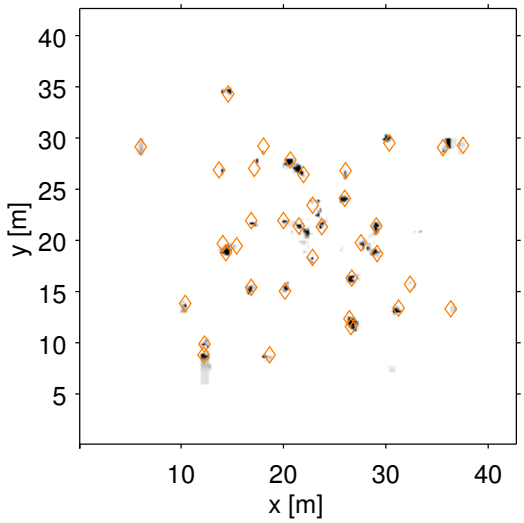
When looking for 10's of targets with three robots, the computational load is relatively small. Assuming that robots move at a constant speed of 0.5 m/s, robots spend an average of 1–3% of the time on filtering updates, 1–5% of the time on control computations, and the remaining time driving, with variations due to the size of the environment which affects the number of particles in the PHD and the number of trajectories. Simulations were run in MATLAB on a desktop computer with an Intel i7 processor and 8 GB of RAM.



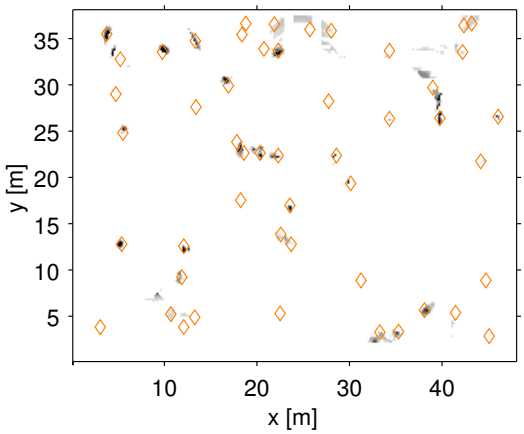
(a) Cardinality – Environment 2



(b) Cardinality – Environment 3

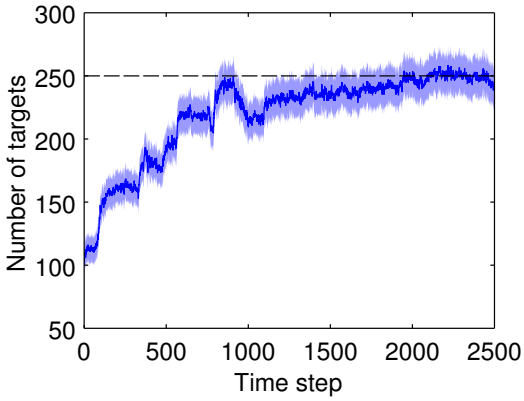


(c) PHD – Environment 2

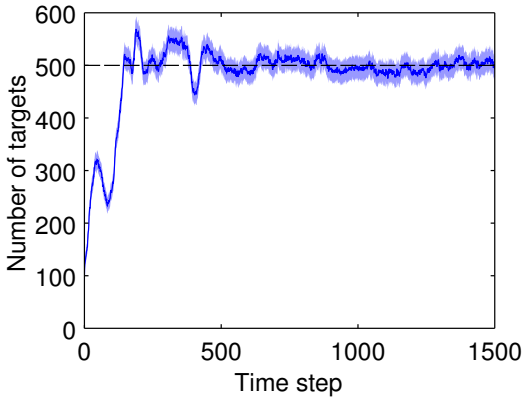


(d) PHD – Environment 3

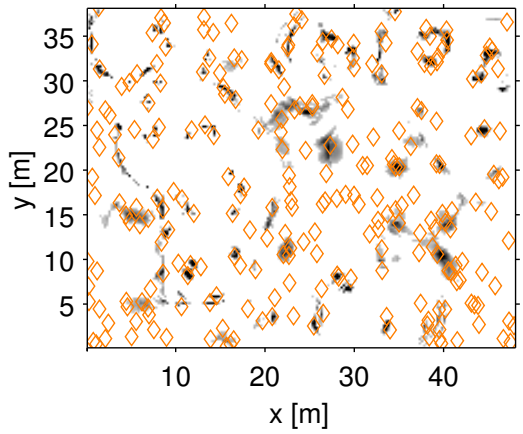
Figure 26: Data showing the expected number of targets over time for environments 2 and 3, where (a) and (b) show the average expected number of targets over time while (c) and (d) show typical final PHD estimates.



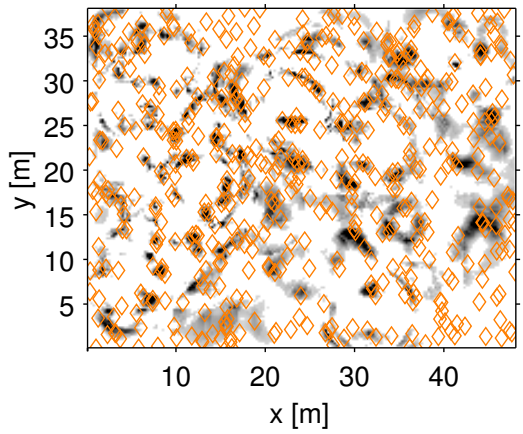
(a) Cardinality – 250 targets



(b) Cardinality – 500 targets



(c) PHD – 250 targets



(d) PHD – 500 targets

Figure 27: Data showing the performance with very large numbers of targets in environment 3.

4.3.3 Large Numbers of Targets

Finally we test the system in situations with an order of magnitude more targets and see that it again performs well, with results shown in Figure 27. While the target cardinality estimates converge to the true number of targets over time in both cases, the target localization is not as good, particularly for targets along the edges of the environment. This is likely due to the combination of a high target density, range-only sensing, and the fact that targets along the edge cannot be viewed from as many positions.

When the number of targets becomes large as in this case, the filter updates take an increasing amount of time, roughly 5–15% of the total time, while the control computations

remain in the range of 1–5% of the total time. This increase in computational time for the filter is due to the increased number of measurements per scan. This could certainly be reduced by optimizing the MATLAB code or switching to `C++`.

4.3.4 Decentralization

The example scenario considered here involves a team of four mobile robots searching for targets within a large indoor office environment, as shown in Figure 28. Robots are equipped with omnidirectional sensors with circular footprints (of radius r_d) and probability of detection given by

$$p_d(\mathbf{x} \mid \mathbf{q}_i) = \begin{cases} p_{d,0} e^{-|\mathbf{x}-\mathbf{q}_i|^2/\sigma_d^2} & \text{if } |\mathbf{x}-\mathbf{q}_i| \leq r_d \\ 0 & \text{if } |\mathbf{x}-\mathbf{q}_i| > r_d \end{cases} \tag{4.28}$$

where $p_{d,0} = 0.8$, $\sigma_d = 2$ m, and $r_d = 5$ m. The measurement model is given by

$$g(\mathbf{z} \mid \mathbf{x}) = \mathbf{x} + \eta \tag{4.29}$$

where $\eta \sim \mathcal{N}(0, \sigma_g^2)$ is Gaussian white noise with $\sigma_g = 1$ m. The expected number of clutter points in the footprint is $\mu = 0.3$ and $c(\mathbf{z}) = \mu/|F|$ is uniform over the footprint.

The robots use two length scales, 1 m and 2 m and plan myopically, *i.e.*, the time horizon $T = 1$ step. The PHD is represented by a set of uniformly spaced particles in a 1 m grid on the robots and a 0.2 m grid on the server and the initial expected number of targets, λ, is set to 20.

There are five access points within the environment and we use a disk model for communication, with access points and robots having a communication range of 10 m. The check-in time, T_C, is set to 40 time steps, well above the minimum number of motions, 23, required to reach any point in the environment from its nearest access point.

Using this setup we simulate the system for 1000 time steps, with the team often finding all the targets and localizing them to within 0.5 m accuracy. To extract the final target estimate from the PHD, we use a simple thresholding and clustering scheme. First, any point with PHD smaller than some $w_{\min} \ll 1$ (we use 0.02) is ignored. From the remaining

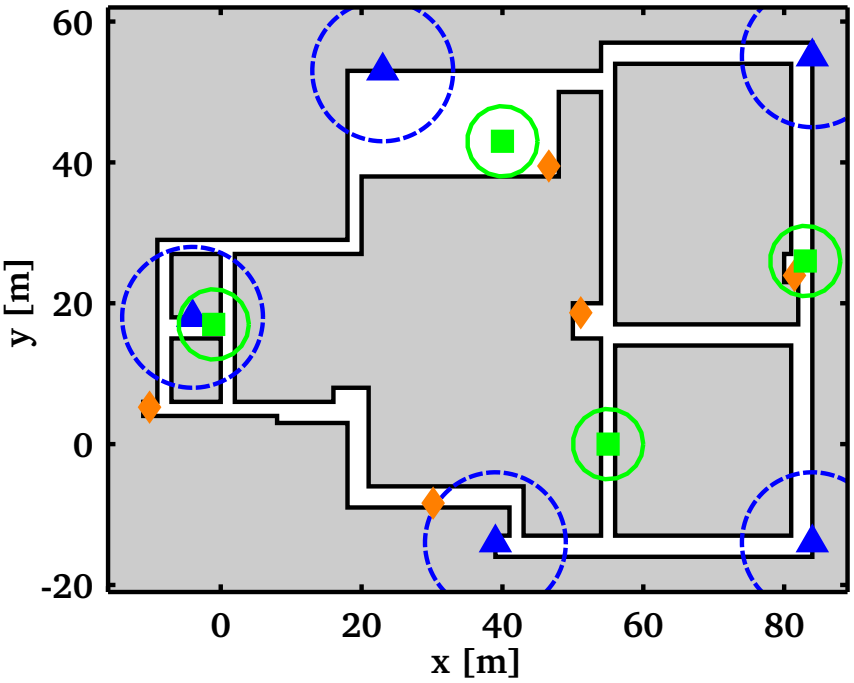


Figure 28: Example environment with four robots (green squares) shown with their sensor footprints (green circles). There are five targets (orange diamonds) and five access points (blue triangles), which have limited communication range (dashed blue circles), within the environment.

points we find clusters with total weight above 0.5, where nodes are connected if they are within an 8-connected neighborhood of one another. Finally, the expected locations are the weighted mean of the particles in each cluster. From a typical trial, the errors in localizing true targets were $0.09, 0.21, 0.29, 0.33, 0.88\,\text{m}$, all less than both the grid size and the standard deviation of the sensor noise. In the same trial there was one false positive target, due to clutter detections while a robot was passing through the hallway in Explore mode with no robot returning to investigate before the simulation ended. Figure 30a shows the time evolution of the control modes for each robot.

4.3.5 Key System Parameters

There are several key parameters that influence the behavior of the robot team. Namely, the number of robots N, the characteristic length of the sensors R_S, the maximum robot velocity V, the number of access points A, the communication range R_C, the check-in time T_C, and the characteristic length of the environment L.

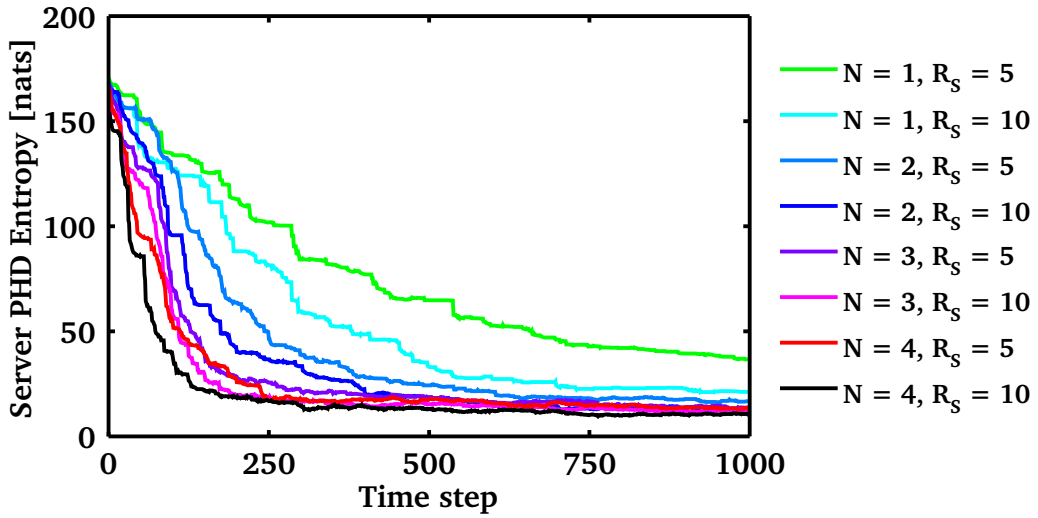


Figure 29: Time evolution of the entropy of the target RFS for a variety of team sizes and footprint radii.

The fraction of information retrieved per time step decreases with the size of the environment, L, but it can be explored more quickly by using more robots, N, or increasing the visible area per robot, R_S. To investigate the effects of N and R_S/L on the rate of information retrieval, we conducted a series of simulations using between 1 and 4 robots and two footprint radii, 5 and 10 m, with 10 trials for each set of parameters. Figure 29 shows the resulting time-evolution of the average entropy (a measure of uncertainty) of the server PHD. As is expected, a higher number of robots and a larger sensing radius both lead to a higher rate of information gathering, as evidenced by the lower entropy.

As the environment grows in size, the time between uploads to the server, T_C, must increase so that robots are able to reach more distant locations. Conversely, robots are able to reach an access point more quickly as the access point density A/L^2, communication range R_C, and robot speed V all increase. To investigate the effects of this exploration time on the system behavior, we ran a series of simulations varying T_C from 10 to 50 time steps by increments of 5, with 10 trials for each rate. Figure 30b shows the average fraction of the total simulation time spent in each control mode. For obvious reasons, it is desirable for the fraction of time spent in the Exploit mode to be as high as possible because this means the robots do not spend large amounts of time driving to access points or getting

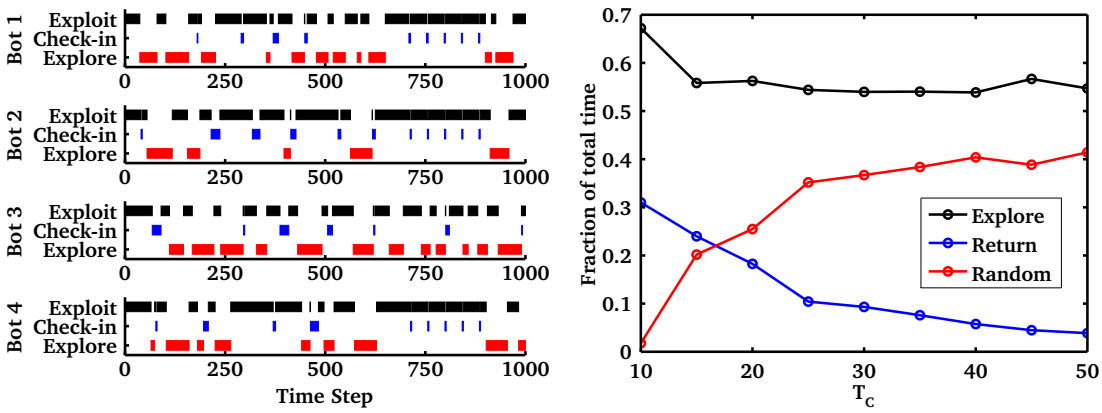
stuck. Not surprisingly, as the check-in rate decreases, the fraction of the total time spent in Check-in mode also decreases. On the other hand, as the ratio of T_C to T_S increases the robot gets stuck more often so it spends more time in Explore mode. The surprising thing is that these two effects appear to cancel one another out, with the total fraction of the times spent exploring at around 0.55 for every value of T_C except $T_C = 10$.

4.3.6 Cooperation

One obvious question to ask is how much benefit leader election within a coalition provides, as opposed to allowing each robot to redundantly plan the coalition action based on its own PHD. In other words, does having different PHDs among the coalition members hurt the performance of the team. To explore this issue we ran another series of simulations where robots did not run the leader election policy. Instead each robot redundantly planned the action of the coalition, effectively acting as the leader but not sharing these plans with other robots.

Figure 31 shows the major difference between the two modes was the rate at which false positive targets arise. While the mean value and standard deviation of true targets are quite similar, the team without the leader election policy has a significantly higher rate of false positive targets. This indicates that one of the primary benefits of leader election is for error mitigation: robots tend to get in each others way or not move in complementary directions when they plan based on different PHDs.

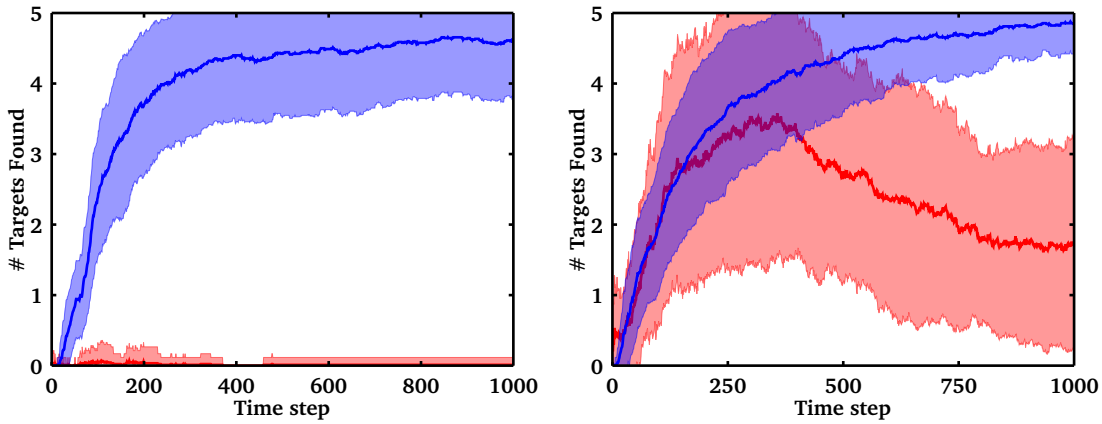
Finally, we return to the issue of computational complexity. In our simulations, run in MATLAB on a laptop with a 2.27 GHz Intel Core i3 with 4 GB of RAM, mutual information for coalition of a single robot took an average of 0.014 s to compute, of two robots an average of 0.484 s, and of three robots an average of 11.829 s. Real-time implementation of this system was not the subject of this work, with these numbers meant to indicate the feasibility, for example using C++ could likely reduce the computation time by an order of magnitude and using a GPU could reduce it significantly more, as mutual information is highly parallelizable. Implementation of the system in hardware will be the study of future work.



(a) Single run

(b) As a function of T_C

Figure 30: Time spent in each control mode, Exploit (black), Check-in (blue), and Explore (red). (a) The time evolution of the mode switching for each individual robot over an example run. (b) The fraction of the total time spent in each mode as a function of T_C.



(a) With leader election

(b) Without leader election

Figure 31: Plots showing the time evolution of the number of true targets (blue) and false targets (red). The mean over 90 separate trials is shown by the solid line and the shaded regions correspond show one standard deviation.

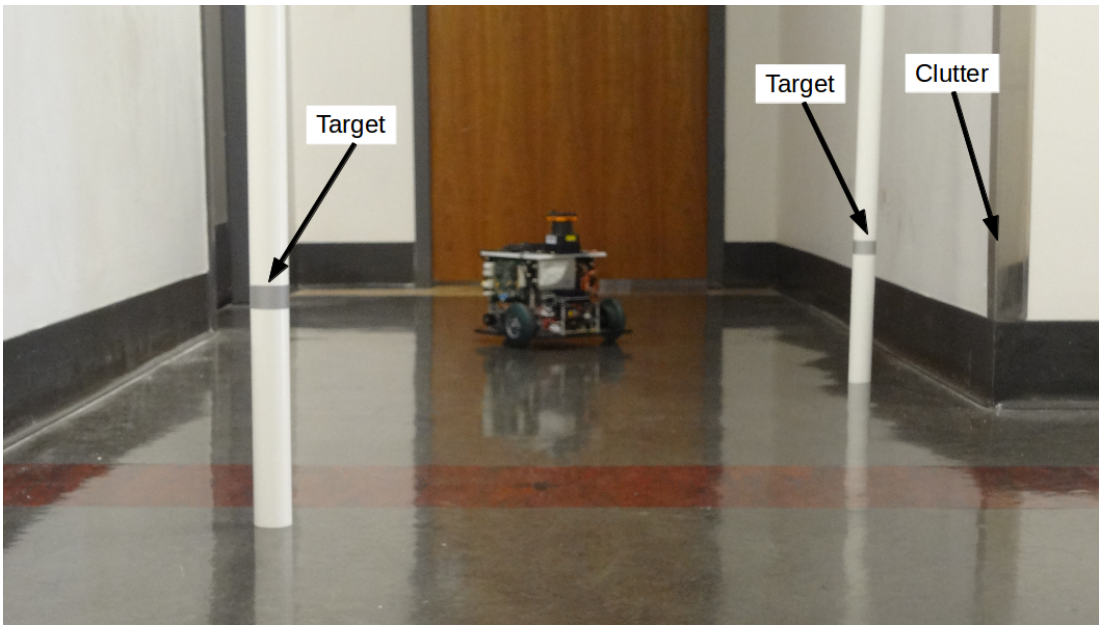


Figure 32: A Scarab robot with two targets in the experimental environment.

4.4 Experimental System and Results

We conduct a series of experiments using a team of ground robots (Scarabs), pictured in Figure 32, to validate the performance of the proposed control algorithm. The Scarabs are differential drive robots with an onboard computer with an Intel i5 processor and 8 GB of RAM, running Ubuntu 12.04. They are equipped with a Hokuyo UTM-30LX laser scanner, used for self-localization and for target detection. The robots communicate with a central computer, a laptop with an Intel i7 processor and 16 GB of RAM running ROS on Ubuntu 12.04, via an 802.11n network. The communication requirements are very modest: individual agents upload their measurement sets, which consist of bearing values, and pose estimates to the central server, and the central server sends out actions to each robot, which consist of a sequence of T poses. The team explores in an indoor hallway in the Levine building, shown in Figure 33, seeking the reflective targets pictured with the robot in Figure 32.

The targets are 1.625 in outer diameter PVC pipes with attached 3M 7610 reflective tape. The tape provides high intensity returns to the laser scanner, allowing us to pick out targets from the background environment. However, there is no way to uniquely identify individual

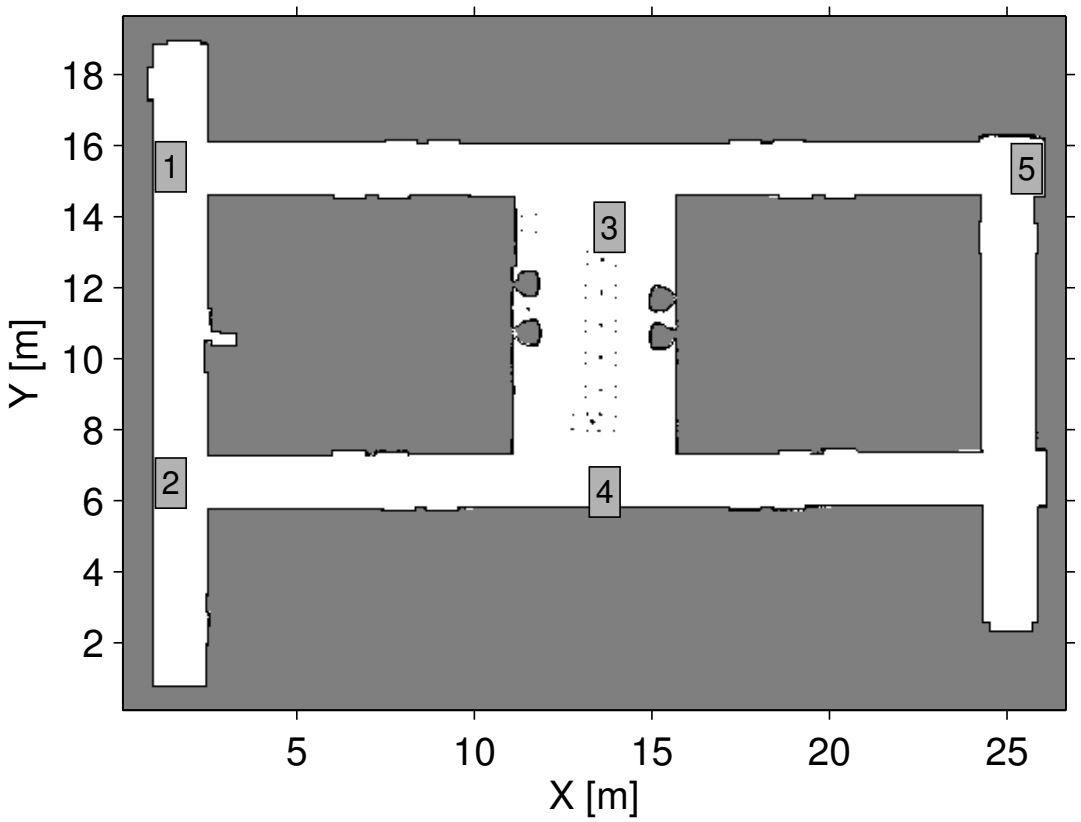


Figure 33: A floorplan of the Levine environment used in the hardware experiments. This map was generated using a manually driven Scarab robot and the `gmapping` package from ROS Gerkey [38]. Different starting locations for the robots are labeled in the map.

targets, making this the ideal setting to use the PHD filter. The hallway features a variety of building materials such as drywall, wooden doors, painted metal (door frames), glass (office windows), and bare metal (chair legs, access panels, and drywall corner protectors, like that in the right side of Figure 32). The reflective properties of the environment vary according to the material and the angle of incidence of the laser. The intensity of bare metal and glass surfaces at low angles of incidence is similar to that of the reflective tape.

We select a threshold on the laser intensity to be able to reliably detect targets within a 5 m range of the robot, at the expense of having occasional false positive detections due to reflective surfaces in the environment. While clutter detections arise due to physical objects within the environment, they are distinct from the targets in one key manner: targets are visible as high-intensity returns from any angle (due to the cylindrical shape and reflective tape) while clutter objects only cause detections when viewed from particular angles.

We converted the Hokuyo laser scanner on-board the robots into a bearing-only sensor, which may be thought of as a proxy to a camera. This simple sensor performs better than a camera in that it avoids common problems such as variable lighting conditions and distortions. Atanasov et al. [4] use the RFS framework to perform semantic self-localization of a robot equipped with a camera, using bearing-only measurements to landmarks. To turn a laser scan into a set of bearing measurements, we first prune the points based on the laser intensity threshold, retaining only those with sufficiently high intensity returns. The points are clustered spatially using the range and bearing information, with each cluster having a maximum diameter d_t. The range data is otherwise discarded. The bearings to each of the subsequent clusters form a measurement set Z.

4.4.1 Sensor Models

We now develop the detection, measurement, and clutter models necessary to utilize the PHD filter. See Appendix B or [23] for further details on the experimental characterization of the sensors.

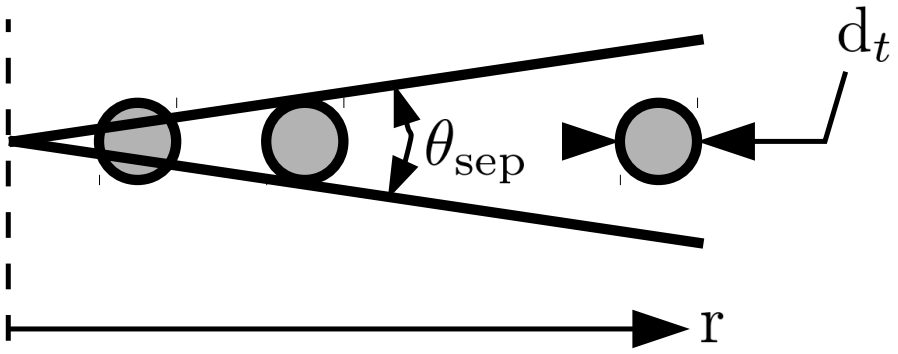


Figure 34: A pictogram of the laser detection model, where d_t is the diameter of the target, θ_{sep} is the angular separation between beams, and r is the range.

Detection Model

The detection model can be determined using simple geometric reasoning due to the nature of the laser scanner, as Figure 34 shows. Each beam in a laser scan intersects a target that is within $d_t/2$ of the beam. The arc length between two beams at a range r is $r\theta_{\text{sep}}$, and the covered space is d_t. Using the small angle approximation for tangent, the probability of detection is

$$p_d(\mathbf{x} \mid \mathbf{q}) = (1 - p_{\text{fn}}) \min\left(1, \frac{d_t}{r\theta_{\text{sep}}}\right) \mathbf{1}\left(b \in [b_{\min}, b_{\max}]\right) \mathbf{1}\left(r \in [0, r_{\max}]\right) \tag{4.30}$$

where r and b are the target range and bearing in the local sensor frame (computed using the robot pose $\mathbf{q}$ and the target position $\mathbf{x}$), p_{fn} is the probability of a false negative, and $\mathbf{1}(\cdot)$ is an indicator function. The bearing is limited to fall within $[b_{\min}, b_{\max}]$ and the range to be less than some maximum value $r_{\max}$ (here due to the intensity threshold on the laser). For our sensor, $b_{\max} = -b_{\min} = \frac{3\pi}{4}$, $r_{\max} = 5\,\text{m}$, $p_{\text{fn}} = 0.210$, and $d_t = 1.28\,\text{in}$. Note that the effective target diameter is less than the true target diameter, since the reflective tape does not provide high intensity returns at extreme angles of incidence.

Measurement Model

The sensor returns a bearing measurement to each detected target. We assume that bearing measurements are corrupted by Gaussian noise with covariance σ, which is independent of

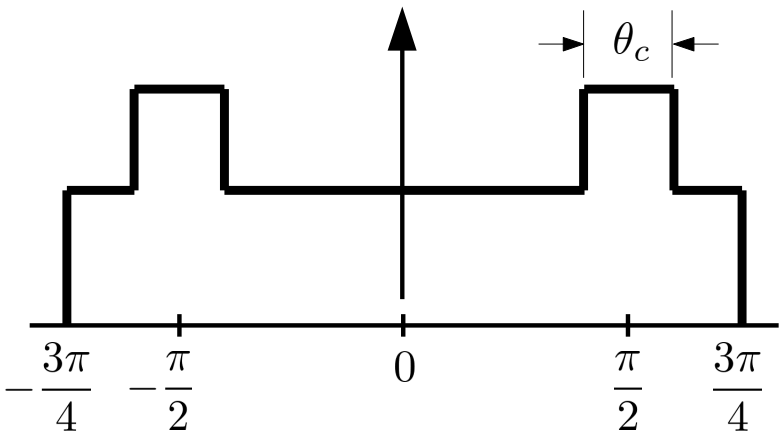


Figure 35: A pictogram of the clutter model, where θ_c is the width of the clutter peaks centered at $\pm\frac{\pi}{2}$, and the bearing falls within the range $[-\frac{3\pi}{4}, \frac{3\pi}{4}]$.

the robot pose and the range and bearing to the target. In other words,

$$g(\mathbf{z} \mid \mathbf{x}, \mathbf{q}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\mathbf{z} - b)^2}{2\sigma^2} \right), \tag{4.31}$$

where b is the bearing of the target in the sensor frame. For our system, $\sigma = 2.25°$.

Clutter Model

As previously noted, clutter (*i.e.*, false positive) measurements arise due to reflective surfaces within the environment, such as glass and bare metal, only at low angles of incidence. For these materials, this most often happens while driving down a hallway, so there will be a higher rate of clutter detections near $\pm\frac{\pi}{2}$ rad in the laser scan. For objects such as metal table and chair legs, there is no clear relationship between the relative pose of the object and robot, so we assume that such detections occur uniformly across the FoV of the sensor. This leads to a clutter model of the form shown in Figure 35.

Let θ_c be the width of the clutter peaks centered at $\pm\frac{\pi}{2}$ and let p_u be the probability that a clutter measurements was generated from a target in the uniform component of the clutter model. The clutter model is

$$c(\mathbf{z}) = \frac{p_u \mu}{b_{\max} - b_{\min}} \mathbf{1}\left(b \in [b_{\min}, b_{\max}]\right) + \frac{(1 - p_u)\mu}{2\theta_c} \mathbf{1}\left(\left||b| - \pi/2\right| \leq \theta_c/2\right), \tag{4.32}$$

where $\mathbf{1}(\cdot)$ is an indicator function and μ is the expected number of clutter measurements per scan. For our system, $b_{\max} = -b_{\min} = 3\pi/4\,\text{rad}$, $p_u = 0.725$, $\theta_c = 0.200\pi\,\text{rad}$, and $\mu = 0.532$.

4.4.2 PHD Filter Implementation Details

The PHD filter is typically implemented as either a weighted particle set, see Vo et al. [102], or a mixture of Gaussians, see Vo and Ma [101]. We use the particle representation as it allows for nonlinear measurement models, such as the bearing-only case described above. Since we assume no knowledge of the initial positions of targets, the particles are initialized with equal weight on a uniformly-spaced grid at all locations at which a target is visible, *e.g.*, in free space for a bearing-only sensor. Ideally, the grid size should be set to a similar length scale as the sensor noise, as below this scale the sensor cannot disambiguate targets. Particles are stationary during the course of the experiment because the targets are stationary. The complexity of the control objective (4.5) is $\mathcal{O}(|Q^{1:R}|2^{RT}(PRT + 2^{RT}))$, where P is the number of particles in the PHD representation.

To reduce the computational complexity of the controller, we subsample the PHD estimate from the filter. We create a uniform grid over the environment, merging all particles within a grid cell into a super-particle with weight equal to the total weight of the merged particles, and position equal to the weighted mean of the merged particles. This is similar to the idea from Charrow et al. [11].

4.4.3 Validation

To evaluate the real-world performance of the proposed algorithm, we run a set of 10 trials in which 3 robots seek 15 targets placed within the office environment from Figure 33. The robots all start near location 1, separated by 0.5 m. The team initially believes there are 30 targets in the environment. The team uses a time horizon of $T = 3$ actions and uses planning mode 1 (concurrently planning over the robots and sequentially over length scales). The robots search over length scales starting at $\ell = 3\,\text{m}$, and increasing by a factor of 1.2 until some robot no longer has any possible destinations due to the limited size of the

environment. With the robots searching over all possible length scales and there being no central server, the robots simply use the Exploit mode from Section 4.2.2. Note that the team typically consists of a single coalition because at long length scales the robots' sensor footprints will overlap. The termination criterion is set to $\epsilon = 0.05$. Figure 36a shows the target cardinality estimates for the team, with the exploration taking 300–500 s to complete.

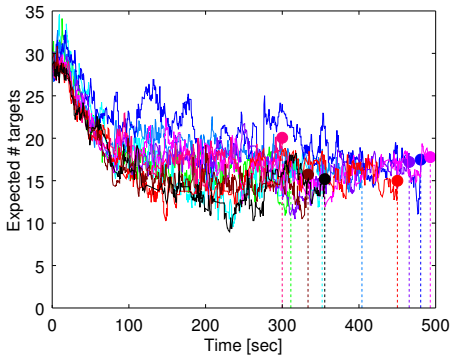
Figure 36b shows the average performance of the team across the trials. The average expected number of targets approaches the true cardinality after approximately 150 s, and stays close for the remainder of the time, showing that the estimator is accurate. The shaded region shows one standard deviation from the mean (where the standard deviation is computed across trials at a given time step) and generally decreases over time, showing that the estimates are also consistent. We scale each run to be of the median time to completion, to be able to compute the standard deviation at a given time instant across runs of different lengths.

Figure 36c shows the true target locations and the localization estimate from a single representative trial. There are 15 unique dots on the map, corresponding to the 15 target locations. The size of the dots is proportional to the expected number of targets at that location. Some targets are better identified than others, *e.g.*, the target in the top middle of the map is larger than some of the other targets. There is also a false positive target near (12, 12) m of low weight compared to the true targets. This false target is due to a cylindrical metal table leg, making it difficult to distinguish from a true target.

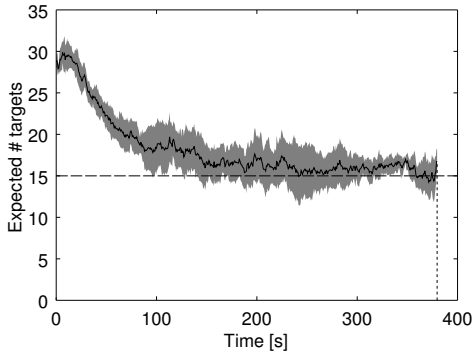
The computational complexity is relatively low, with control actions taking an average of 1.01 s, and a maximum of 3.37 s, to compute. The team spent 4.7% of the total exploration time stationary, planning their next actions: a small, but not negligible, fraction of the time.

4.4.4 Team Size Comparison

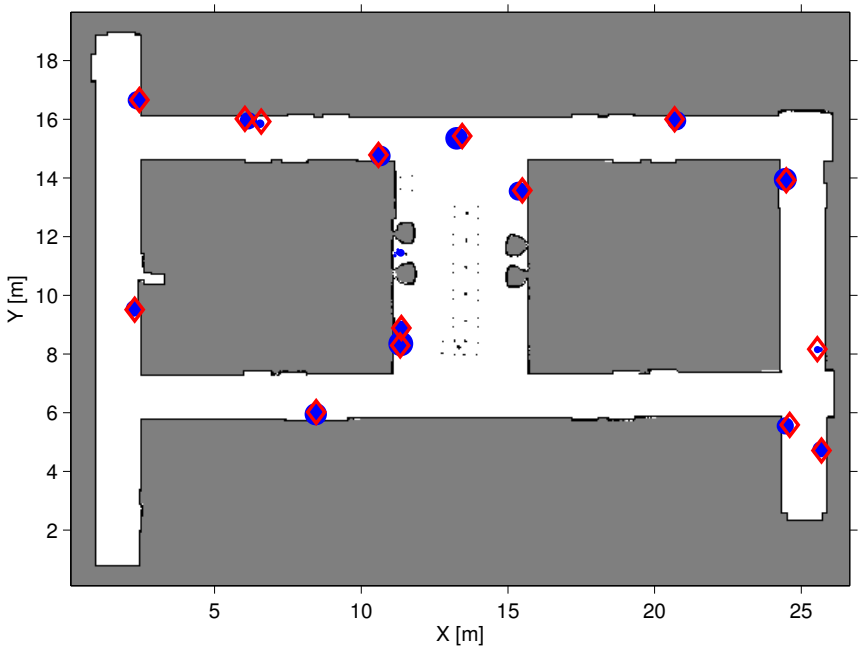
We conduct a series of 10 trials each with 1, 3, and 5 robots in the same environment as above to explore the effect of team size on the exploration performance. For the 5 robot trials, one robot starts at each of the labeled locations in Figure 33; for the 3 robot trials they begin at locations 1–3; and for the single robot trials it begins at location 3. The robots use



(a) Expected cardinality over time



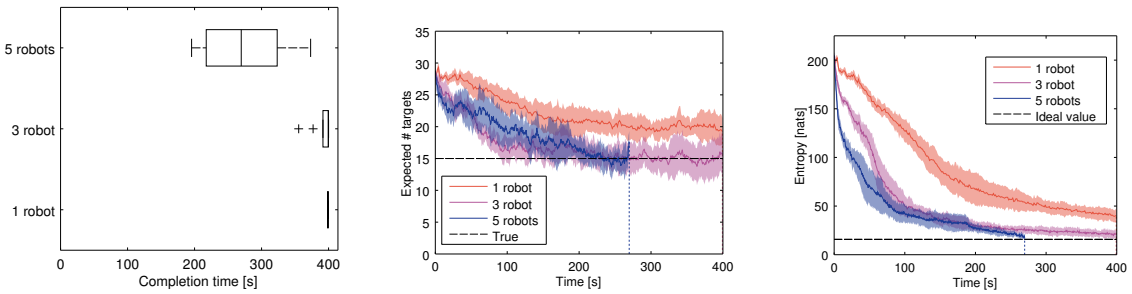
(b) Average expected cardinality as a function of normalized time



(c) Estimated target locations

Figure 36: Plots of the performance of a team of three real-world robots exploring the Levine environment using planning mode 1. (a) Shows the expected cardinality of the team over time, with the final cardinality in each run marked by a circle and the final time as a dotted vertical line. (b) Shows the mean (solid black line) and standard deviation (shaded region) of the expected cardinality across runs over time, with the true cardinality shown (dashed black line). (c) Shows the true (red diamonds) and estimated target locations (blue dots), with the dot size proportional to the estimated number of targets at that location.

planning mode 3 (sequentially over both robots and length scales), as concurrently planning over robots is prohibitively expensive for 5 robots. All of the other parameters are identical to the previous trials.



(a) Completion times

(b) Average expected cardinality as a function of time

(c) Average target set entropy as a function of time

Figure 37: Plots of the performance for teams of 1, 3, and 5 real-world robots exploring the Levine environment using planning mode 3. (a) Shows the spread of time to completion. (b) Shows the mean (solid lines) and standard deviation (shaded regions) of the expected cardinality across runs over time budget, with the true cardinality shown (dashed black line). (c) Shows the mean (solid lines) and standard deviation (shaded regions) of the entropy across runs over time with the ideal value shown (dashed black line).

The team is given a time budget of 400 s to complete the exploration task, with Figure 37a showing the statistics of the completion times. Within the time budget, the single robot never completes the task, the three robot team completes it in 3 of the 10 runs, and the 5 robot team completes it every time, in a median of 270 s and a maximum of 373 s. As is expected, adding more robots improves completion time, as they are able to simultaneously gather measurements from more locations than a smaller team. Figure 37b and Figure 37c show the average cardinality estimates and target set entropies for each of the team sizes. The 5 robot trials have the highest rate of entropy reduction and the 3 robot teams nearly finish exploring the environment, with the entropy approaching the terminal value at the end of the trials. The single robot case is furthest from convergence, with the final entropy at the same level that a 3 robot team achieves in 34% of the time.

With planning mode 3, the computational load is minor: taking an average of 0.029 s for 1 robot, 0.092 s for 3 robots, and 0.351 s for 5 robots. This is 0.18% of the total time for 1 robot, 0.42% for 3 robots, and 1.45% for 5 robots, all less than the Mode 1 planning in the previous experiments. However, mode 3 is not guaranteed to return plans with as high of an expected information gain as mode 1.

4.5 Simulation Results

We also conduct a series of simulation experiments to further explore the performance of the proposed control strategy (4.5), varying the planning method, target density, environment, and sensing modality.

4.5.1 Simulator Validation

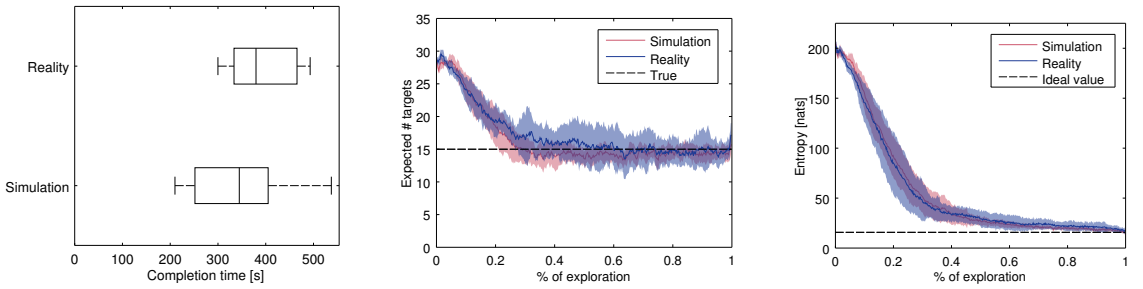
We wish to verify that the simulation environment behaves similarly to the experimental system before conducting a long series of trials in simulation. To do this, we mimic the setup from Section 4.4.3 as closely as possible, using identical sensor parameters, controller parameters, team size, planning method, etc. The target locations for the simulation are set to the true locations shown in Figure 36c.

Overall, the results in Figure 38 show that the two systems are comparable, with both systems able to accurately and consistently estimate the target set cardinality and reach the desired level of confidence in their estimate. The experimental data is more consistent across runs, both in terms of completion time and for inter-run estimates of the target set cardinality and entropy. However, the simulated system has a lower median time of completion, at 338 s compared to 392 s. While there are some differences, overall the systems are similar enough to trust that further simulated results will not differ significantly from experimental results.

4.5.2 Planning Method Comparison

In Section 4.4, we use planning modes 1 and 3, but could not make direct comparisons between the two due to the different team setups. We now wish to see how the different planning methods affect the team's performance, and verify that taking intelligent actions (*i.e.*, maximizing mutual information) outperforms a naïve random walk. In theory, mode 0 leads to plans with the highest expected information gain, but the plans would take longer to compute, potentially causing the actual information gain over time to decrease. Modes 1 to 3 are all approximations, sequentially planning over the length scales, team members, or both, and robots using modes 4 and 5 randomly select actions.

We use the same setup as Section 4.4.3, but vary the planning modality and set a time



(a) Completion times

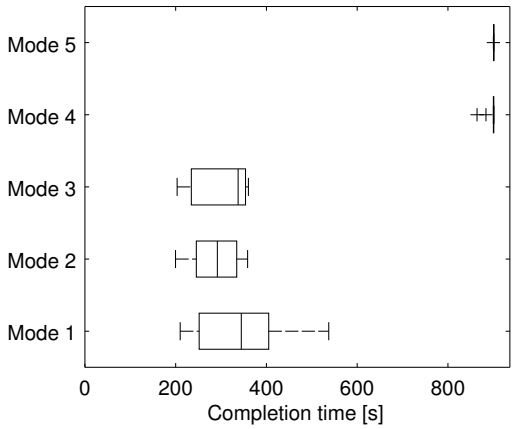
(b) Average expected cardinality as a function of normalized time

(c) Average target set entropy as a function of normalized time

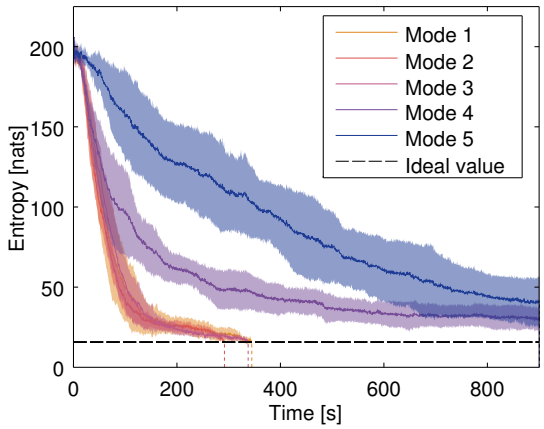
Figure 38: Plots of the performance for teams of three real and simulated robots exploring the Levine environment using planing mode 1. (a) Shows the spread of time to completion. (b) Shows the mean (solid lines) and standard deviation (shaded regions) of the expected cardinality across runs as a fraction of the total time with the true cardinality shown (dashed black line). (c) Shows the mean (solid lines) and standard deviation (shaded regions) of the entropy across runs as a fraction of the total time with the ideal value shown (dashed black line).

budget of 900 s. Figure 39 shows the results of the trials. Information-based control of any kind significantly outperforms the random walk in terms of completion time and estimation accuracy. The information-based methods (modes 1–3) all converge to the desired target set entropy in all of the trials, mode 4 (random walk with concurrent length scales) completes the exploration before the time budget expires in 2 of the 10 trials, and mode 5 (random walk with sequential length scales) never completes the task. Mode 5 is also much less consistent than all of the other modes in terms of the rate of entropy reduction.

It is not surprising that mode 2 (planning sequentially over robots and concurrently over length scales) has the lowest median completion time and that the spread is narrower compared to modes 1 and 3. When robots are allowed to plan over different length scales, some robots explore local regions of high uncertainty while other robots move across the environment to search for new targets, allowing the team to more efficiently explore. Using modes 1 and 3, all robots act at the same length scale so some robots must occasionally act on a undesirable length scale for the benefit of other team members. It is surprising that mode 1 leads to the most inconsistent completion times, as the expected information gain is an upper bound for the gain in mode 3. The differences could have been due to chance,



(a) Completion times



(b) Average target set entropy as a function of time

Mode	Avg. time	Max time	% total time
1	1240	1790	9.50
2	162	188	0.86
3	211	246	1.63
4	0.3	0.4	0.002
5	0.8	0.9	0.01

(c) Table of computation times in ms

Figure 39: Plots of the performance for a team of three simulated robots exploring the Levine environment using planning modes 1–5. (a) Shows the spread of time to completion. (b) Shows the mean (solid lines) and standard deviation (shaded regions) of the entropy across over time budget with the ideal value shown (dashed black line). (c) Shows the computation times in ms, and the percentage of the total time spent computing.

as there are only 10 trials with each planning modality.

Both of the random exploration methods (modes 4 and 5) perform significantly worse than the information-based planning. Not only does mode 5 never complete the exploration in the given time budget, its rate of entropy reduction is significantly slower than all of the other methods, including mode 4. While the planning times for the random methods are negligibly small, as Figure 39c shows, this does not counteract the fact that the actions are not being selected in an intelligent manner.

4.5.3 Target Cardinality Comparison

We next test the performance of the system in situations with variable target cardinalities, and, correspondingly, variable target densities in the environment. The PHD filter functions for any target cardinality and the exploration controller is agnostic to the target cardinality. We conduct a series of experiments with the same setup, except we use 1, 15, and 100 targets in the Levine environment, shown Figure 33, which is approximately 144 m^2. Three robots explore, starting at location 1 in the map and using planning mode 2.

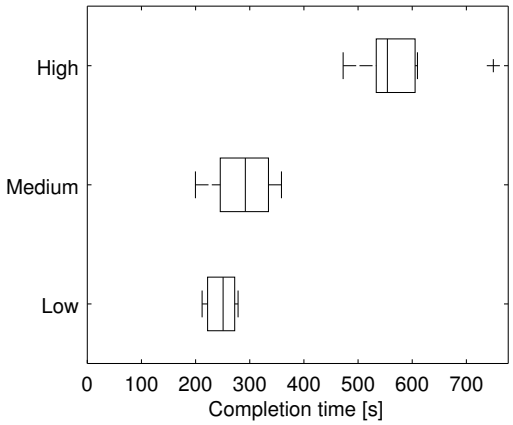
Figure 40 shows the completion times, cardinality estimates, and target set entropies. As expected, the low target density is the fastest to complete, since the team simply needs to sweep out the mostly empty space. With the high target density the robots need to observe the many targets from a variety of vantage points to localize them with sufficient confidence, a process which involves sweeping across the environment multiple times. The variability in completion time, the time to correctly estimate the true cardinality, and the inconsistency of the cardinality estimates also increases with the target cardinality. In fact, on average, the high target cardinality runs do not reach the correct cardinality until about 95% of the way through exploration.

Figure 40c shows that the team reaches the desired level of uncertainty at the end of each run. Note that for the high cardinality runs, the initial rate of target discovery is higher than the initial rate of target localization, resulting in an increase in entropy for the first 40 s of the run.

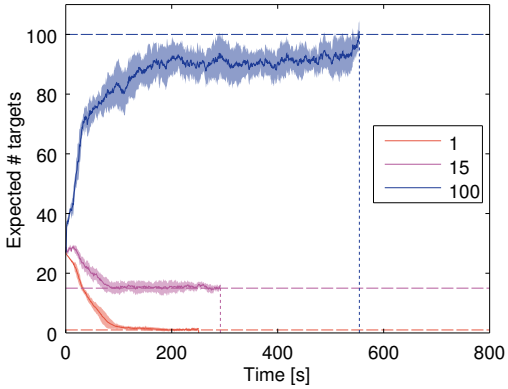
4.5.4 Second Environment

We next conduct a series of simulations in the larger, more complex indoor environment shown in Figure 41a. This environment features many rooms for the robots to explore, and is nearly four times the area of the Levine environment. There are 40 targets in the environment, and the termination criterion is increased to $\epsilon = 1$. Three robots begin in room 1 in the map and use planning mode 2. All of the other system parameters are identical.

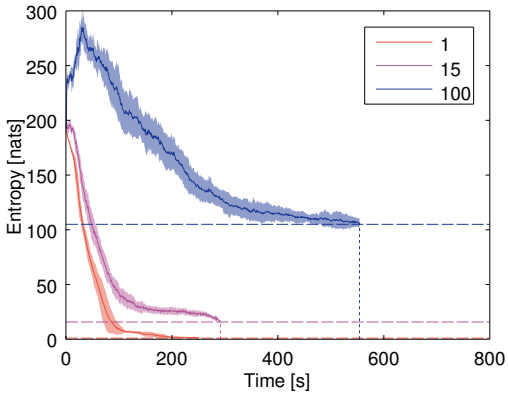
Figure 41 shows that the team is able to accurately and consistently estimate the true



(a) Completion times



(b) Average expected cardinality as a function of normalized time



(c) Average target set entropy as a function of normalized time

Figure 40: Plots of the performance for a team of three simulated robots exploring the Levine environment for 1, 15, or 100 targets using planning mode 2. (a) Shows the spread of time to completion. (b) Shows the mean (solid lines) and standard deviation (shaded regions) of the expected cardinality across runs as a fraction of the total time with the true cardinality shown (dashed black line). (c) Shows the mean (solid lines) and standard deviation (shaded regions) of the entropy across runs as a fraction of the total time with the ideal value shown (dashed black line).

target cardinality and reach the desired level of confidence in the target estimate. The robots take significantly longer to complete the exploration compared to the Levine environment, with a median time of 2220 s, 6.56 times as high. We expect the time to be at least 4 times as high due to the increase in area, with the extra time likely caused by the increased complexity, as the robots must enter many individual rooms.

Since the environment is larger, there are more length scales for the robots to consider, and thus more actions. This increases the planning time. Planning mode 2 takes an average of 1.63 s per plan, and in total is 9.2% of the exploration time.

4.5.5 Range-Only Sensing

Nothing about the estimation or control framework relies upon the sensor modality, so long as we are able to create detection, measurement, and clutter models for the sensor. To verify this, we conduct a final series of simulation experiments in which robots are equipped with noisy, range-only sensors.

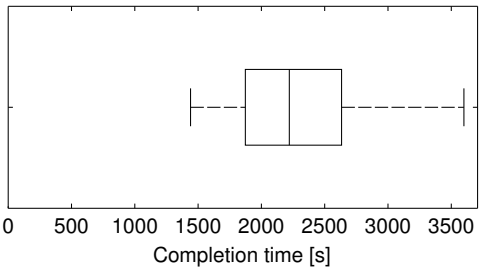
Sensor Models

The range-only sensor parameters used in this case are not based on a particular physical sensor, but rather seek to capture the general behavior of an RF-based range sensor. Figure 64 shows the detection model for the sensors, $p_d(\mathbf{x} \mid \mathbf{q})$, which decays steadily with distance. The measurements have zero-mean Gaussian noise, so $\mathbf{z} \sim \mathcal{N}(|\mathbf{x} - \mathbf{q}|, \sigma^2)$, where $\sigma = 1\,\text{m}$. The measurement noise is relatively high compared to the bearing-only sensor, so we expect the rate of information gain to be lower. Clutter detections occur uniformly over the sensor footprint, with a clutter PHD $c(\mathbf{z}) = \mu / r_{\max}$, where $\mu = 0.1$ is the clutter rate and $r_{\max} = 5\,\text{m}$ is the maximum range of the sensor.

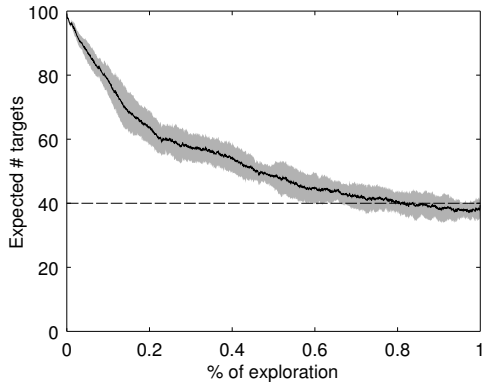
Results

Most of the simulation parameters are kept constant: a team of 3 robots begin at location 1 in Levine, and use planning mode 2 with the same length scales as the bearing-only sensor. The termination criterion is $\epsilon = 4$ to account for the much coarser localization that the range-only sensor is able to achieve, due to the high measurement noise.

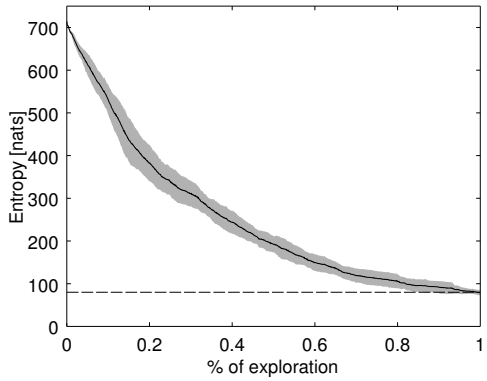
(a) Floorplan



(b) Completion times



(c) Average expected cardinality as a function of normalized time



(d) Average target set entropy as a function of normalized time

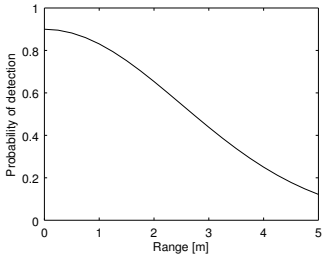
Figure 41: Plots of the performance for a team of three simulated robots exploring a second environment using planning mode 2. (a) Shows the floorplan of the complex, indoor environment used in simulations. The robots begin in room 1 in the upper left corner. (b) Shows the spread of time to completion. (c) Shows the mean (solid lines) and standard deviation (shaded regions) of the expected cardinality across runs as a fraction of the total time with the true cardinality shown (dashed black line). (d) Shows the mean (solid lines) and standard deviation (shaded regions) of the entropy across runs as a fraction of the total time with the ideal value shown (dashed black line).

Figure 42 shows the resulting completion times, cardinality estimates, and target set entropies, as well as an example localization result. As is expected, the system takes longer to complete the localization task, and the resulting target estimates are not as precise as with the bearing-only sensor. The team is able to discover the approximate locations of all of the targets, though there are a number of false positive targets that appear in the final PHD estimate. In particular, it is difficult for the team to eliminate the false target near $(15,\ 18)\,\mathrm{m}$ as this is only observable from several meters below and there are true targets nearby at very similar ranges to the false target. Despite the errors in target localization, the team is still able to accurately estimate the true target cardinality.

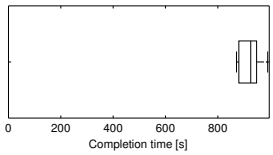
4.6 Conclusion

In this chapter, we proposed a novel receding-horizon, information-based controller for actively detecting and localizing an unknown number of targets using a small team of autonomous mobile robots. The robots are equipped with unreliable sensors, failing to detect targets within the field of view, returning false positive detections, and being unable to uniquely identify true targets. Despite this, the PHD filter simultaneously estimates the number of targets and their locations, avoiding the need to explicitly consider data association and providing a scalable approach for various team sizes, sensor modalities, and environments.

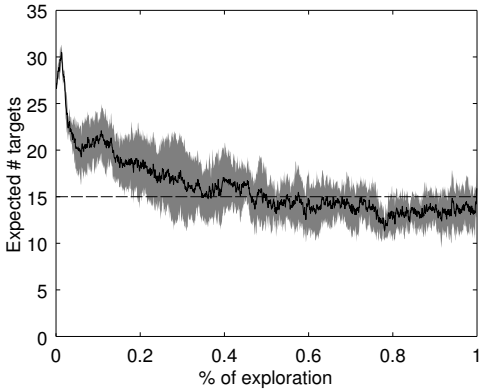
The controller, which maximizes the mutual information between the target set and the future binary measurements of the team, hedges against highly uninformative actions in a computationally tractable manner. We provide several variations on the controller: concurrently or sequentially planning across robots in the team and length scales of actions, planning in a decentralized fashion, and comparing the performance of the PHD and CPHD filters. The PHD filter, somewhat counterintuitively, outperforms the CPHD filter when the sensor has a finite footprint, with the CPHD filter performing poorly in terms of the cardinality estimation. The effectiveness of our control strategy is demonstrated through a series of hardware experiments with small teams of ground robots exploring an indoor office



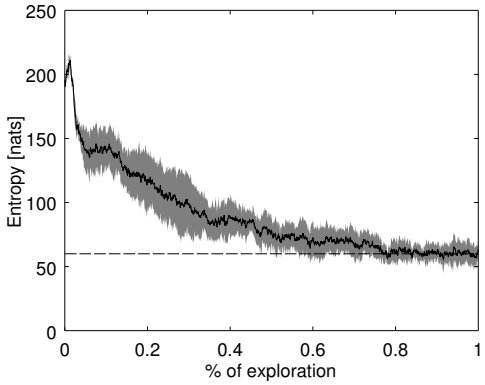
(a) Range-only detection model



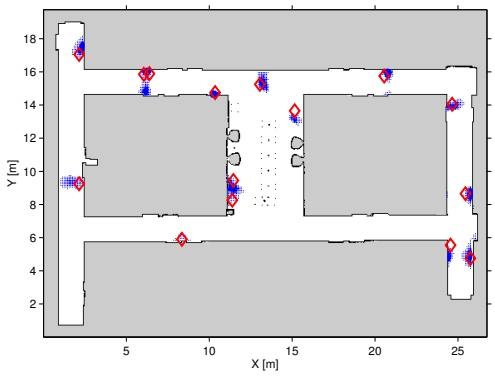
(b) Completion times



(c) Average expected cardinality as a function of normalized time



(d) Average target set entropy as a function of normalized time



(e) Estimated target locations

Figure 42: Plots of the performance for a team of three simulated robots equipped with range-only sensors exploring the Levine environment using planning mode 2. (a) Shows the detection model used in the simulation trials. (b) Shows the spread of time to completion. (c) Shows the mean (solid lines) and standard deviation (shaded regions) of the expected cardinality across runs as a fraction of the total time with the true cardinality shown (dashed black line). (d) Shows the mean (solid lines) and standard deviation (shaded regions) of the entropy across runs as a fraction of the total time with the ideal value shown (dashed black line). (e) Shows the true (red diamonds) and estimated (blue dots) target locations, with dot size proportional to the number of targets at that location.

environment. A series of simulated experiments show that the proposed approach performs well in a variety of settings: with low and high target cardinality, in multiple environments, and with multiple sensor modalities. The proposed control law also significantly outperforms a random walk through the environment without significantly increasing the computational load. The team is able to autonomously cease exploration once their confidence in the target estimates is sufficiently high.

Chapter 5

Active Detection, Localization, and Tracking of Moving Targets

Target tracking is a fundamental problem in robotics research and has been the subject of detailed studies over the years. In this chapter, consider the problem of tracking an unknown and dynamic number of mobile targets with a team of robots. We present a greedy algorithm for assigning trajectories to the robots that maximize submodular objective functions and prove that this is a 2-approximation. We examine two such objective functions: the mutual information between the estimated target positions and future measurements from the robots, and the expected number of targets detected by the robot team. We provide extensive simulation evaluations using a real-world dataset. The research in this chapter was originally published in [28].

5.1 Introduction

Target detection, localization, and tracking has many applications including search-and-rescue [36], wildlife tracking [98], surveillance [42], and building smart cities [63]. Consequently, such problems have long been a subject of study in the robotics community. Target tracking typically refers to two types of tasks: estimating the trajectories of the targets from the sensor data, and actively controlling the motion of the robotic sensors to gather

the data. We address both types of problems for the case of multiple, moving targets.

Unlike most existing work, we study the case of tracking an *unknown* and *varying* number of *indistinguishable* targets. This introduces a number of challenges. First, we cannot maintain a separate estimator for each target, since the required number of estimators is unknown. Second, we must account for the fact that targets appear and disappear from the environment. Third, we cannot maintain a history of the target positions because we cannot uniquely identify individual targets, making prediction difficult. Finally, the system must be capable of handling false positive and false negative detections and unknown data association in addition to sensor noise. Despite these challenges, we present positive results towards solving the problem.

An important consideration for target tracking is the motion model for the targets. A number of parametric motion models have been proposed in the literature (see [61] for a detailed survey). We employ a data-driven technique to extract the motion model, instead of assuming any parametric form. Specifically, we use Gaussian Process (GP) regression to learn a map of velocity vectors for the targets, similar to Joseph et al. [49]. Additionally, we show how to model the appearance and disappearance of targets within the environment. Next, we present a control policy to assign trajectories for all robots in order to maximize the objective function over a receding horizon. We study two objective functions using the PHD filter: mutual information and the expected number of detections by the robots. We show that both objective functions are submodular, and use a result based on [99] to prove that our greedy control policy is a 2-approximation.

In addition to the theoretical analysis we offer, we evaluate our algorithm using simulated experiments. While our framework may be applied to a number of robot and sensor models, for the purposes of testing we restrict our attention to fixed winged aerial robots with downward facing cameras. We use a real-world taxi motion dataset from [80] for the targets and to verify our models. The simulation results reveal that robot teams using the information-based control objective track a smaller number of targets with higher precision compared to teams that maximize the expected number of detections.

5.1.1 Related Work

Active target tracking problems have been studied in the literature under many different settings. Solutions have been presented for radio-based sensors [46], range-only sensors [108], bearing sensors [64], and range and/or bearing sensors [109], under centralized and decentralized settings. Frew and Rock [34] design optimal trajectories for a single robot to track a single moving target using monocular vision. The problem of keeping targets in a robot's field-of-view can be formulated as a visual servoing problem. Gans et al. [37] design a controller which guarantees stability while keeping three or fewer targets in the field-of-view of a single mobile robot.

[94] present a general solution for the multi-robot, multi-target case using a particle filter formulation. Tracking multiple targets with multiple robots requires explicit or implicit assignment of targets to robots. Xu et al. [106] present a mixed nonlinear integer programming formulation for assigning robots to targets as well as for determining optimal robot positioning. Such a formulation is not directly applicable in our case since the number of targets itself is unknown, and thus explicit assignment is not possible. Tokekar et al. [99] present a greedy tracking algorithm for a team of aerial robots. In this chapter, we build on this work to allow for the case of an unknown and changing number of targets. Recently, there has been some work on actively detecting and/or localizing an unknown number of targets using radio sensors [54, 92], range-only sensors [12], and arbitrary sensor models [25]. Kim et al. [54], Song et al. [92] studied the problem of detecting and localizing an unknown number of radio sources. Unlike all these works, we do not assume that the targets remain stationary.

5.2 Problem Formulation

We address the problem of a team of R robots monitoring an area E in order to detect, localize, and track an unknown number of moving targets using an inexpensive camera. The robots are able to localize themselves within the environment (*e.g.*, using GPS) and robot r has pose $\mathbf{q}_t^r$ at time t.

The number of targets, n_t, is unknown and varies over time, since individual targets may enter and leave the area of interest. We use Random Finite Sets (RFSs) to represent the number and state of targets at any time. Let $X_t = \{\mathbf{x}_{1,t}, \mathbf{x}_{2,t}, \ldots, \mathbf{x}_{n_t,t}\}$ denote a realization of an RFS of target states at time t. Each robot receives a set of measurements $Z_t^r = \{\mathbf{z}_{1,t}^r, \mathbf{z}_{2,t}^r, \ldots, \mathbf{z}_{m,t}^r\}$ to targets that it detects within the field of view (FoV) of its sensor. The number of measurements, m_t, varies over time due to false negative and false positive detections and the motion of the robots and the targets. Let $p_d(\mathbf{x} \mid \mathbf{q})$ denote the probability of a robot at $\mathbf{q}$ detecting a target with state $\mathbf{x}$. For targets outside of the FoV of the sensor, $p_d(\mathbf{x} \mid \mathbf{q}) \equiv 0$. Here, $p_d(\mathbf{x} \mid \mathbf{q}) \in (0, 1)$ indicates the possibility of a false negative, or missed, detection. When a target is successfully detected, the sensor will return a measurement $\mathbf{z} \sim g(\cdot \mid \mathbf{x}, \mathbf{q})$. The sensor can also return measurements to clutter objects, causing false positive detections. Let $c(\mathbf{z} \mid \mathbf{q})$ denote the PHD of clutter measurements.

5.3 Target Tracking Framework

The representative problem that we consider is of a team of fixed-wing aerial robots equipped with downward-facing cameras tracking vehicles driving on the ground. However the same methodology could be extended to work with robots with other mobility constraints (*e.g.*, ground vehicles or quadrotor platforms) and other sensor modalities (*e.g.*, lidars or 3D depth cameras).

5.3.1 Sensor Parameterization

The problem of detecting vehicles using aerial imagery has been well studied [39, 107]. We use such studies to inform our selection of the sensor detection, measurement, and clutter models. The approaches presented in [39, 107] are similar, searching for image features over a range of scales in order to detect cars of different sizes or to detect cars from different elevations or with different image resolutions. In general, the system is able to have a higher detection rate if we accept a larger number of false positive detections [107, Fig. 12], [39, Fig. 8]. The detection rate may also vary with the number of pixels per target, which may be computed, using the robot pose, the approximate length scale of a target, and the image

resolution, to be

$$\#\text{ pixels per car} = \text{pixels per radian} \times \arctan \frac{\text{length of target}}{\text{distance from camera to target}}. \tag{5.1}$$

We assume a logistic relationship between the number of pixels per target, $n_{\text{px}}(\mathbf{x}, \mathbf{q})$, and the detection rate,

$$p_d(\mathbf{x} \mid \mathbf{q}) = p_0 + \frac{p_{d,\max} - p_{d,0}}{1 + \exp\big(- k(n_{\text{px}}(\mathbf{x}, \mathbf{q}) - n_{p,0}) \bigcap \big)}, \tag{5.2}$$

where $p_{d,0}$, $p_{d,\max}$, k, and $n_{px,0}$ are design parameters.

The camera returns pixel (*i.e.*, bearing) measurements to the cars detected within the image. Using the pose of the robot, we can project measurements onto the ground plane to localize the targets. The measurement model is

$$g(\mathbf{z} \mid \mathbf{x}, \mathbf{q}) = \mathcal{N}(\mathbf{z}; [r_{\mathbf{x}}, c_{\mathbf{x}}]^T, \sigma^2 I), \tag{5.3}$$

where $r_{\mathbf{x}}, c_{\mathbf{x}}$ are the pixel row and column values in an image taken at $\mathbf{q}$, of a target at $\mathbf{x}$, σ is the standard deviation in pixels, and I is a 2×2 identity matrix, as we are tracking the two-dimensional position of the targets.

Like the targets, the clutter is modeled as a Poisson RFS, which is completely characterized by the PHD. Without *a priori* knowledge of locations that are likely to have clutter, the best choice is to use a uniform distribution over the measurement space. For most computer vision-based detection algorithms, the expected number of clutter detections depends upon the detection model, with a high detection likelihood resulting in a higher detection rate [39, 107].

5.3.2 Target Parameterization

In order to predict how the target set evolves, we need models for the motion of individual targets as well as the birth/death processes of the targets. A number of motion models have been proposed in the literature, ranging from adversarial [17] to stochastic [61]. Often, a

mixture of parametric motion models is used [62]. We take a data-driven approach to modeling the targets' motion, utilizing real-world datasets that are available [55]. In particular, we use Gaussian Process (GP) regression [83] to learn the function that maps the position coordinates of the targets to velocity vectors, as shown by Joseph et al. [49].

GP regression is a Bayesian approximation technique to learn some function $f(X)$ given measurements $y = f(\mathbf{x}) + \epsilon$ corrupted by Gaussian noise, $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Here, $\mathbf{x} = [x_1, x_2]^T$ refers to the position coordinates of the targets. We learn two separate functions, f_1 and f_2, one for each axes of the ground plane, assuming that the velocities along the two axes are independent. Instead of assuming a parametric model for f_i, GP regression assumes that the joint distribution of $f_i(X)$ defined over any collection of positions, $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$, is always Gaussian. Thus, $f_i(X)$ is completely specified by its mean function, $m_i(X) = E[f_i(X)]$ and covariance function, $k_i(X, X') = E[(f_i(X) - m_i(X))(f_i(X') - m_i(X'))]$.

Given observed velocity vectors $\mathbf{y}_1$ and $\mathbf{y}_2$ taken at some subset of positions, X, GP regression predicts the velocity vectors at another set of positions, X^*, as a Gaussian distribution with conditional mean and variance values [83]:

$$m_i(X^*|X) = m_i(X^*) + K_i(X^*, X)[K_i(X, X) + \sigma^2 I]^{-1}(Y_i - m_i(X))$$
$$\sigma_i^2(X^*|X) = K_i(X^*, X^*) - K_i(X^*, X)[K_i(X, X) + \sigma^2 I]^{-1}K_i(X, X^*),$$

where $K_i(X, X')$ is a matrix whose $(m, n)^{th}$ entry is given by the covariance between $\mathbf{x}_m \in X$ and $\mathbf{x}_n \in X'$. We take the prior function, $m_i(X)$, to be a zero-mean distribution. Thus, if the covariance function is known, the above equations can fully predict the velocity values at arbitrary positions.

We assume that the covariance function belongs to the Matérn class with parameter $\nu = 3/2$ [83] since this choice of covariance function yields a better fit as compared to the standard squared-exponential function used by Joseph et al. [49]. The length hyperparameter of the Matérn covariance is learned using training data from the Cabspotting taxi dataset from [80]. The training dataset consists of time-stamped GPS latitude and longitude

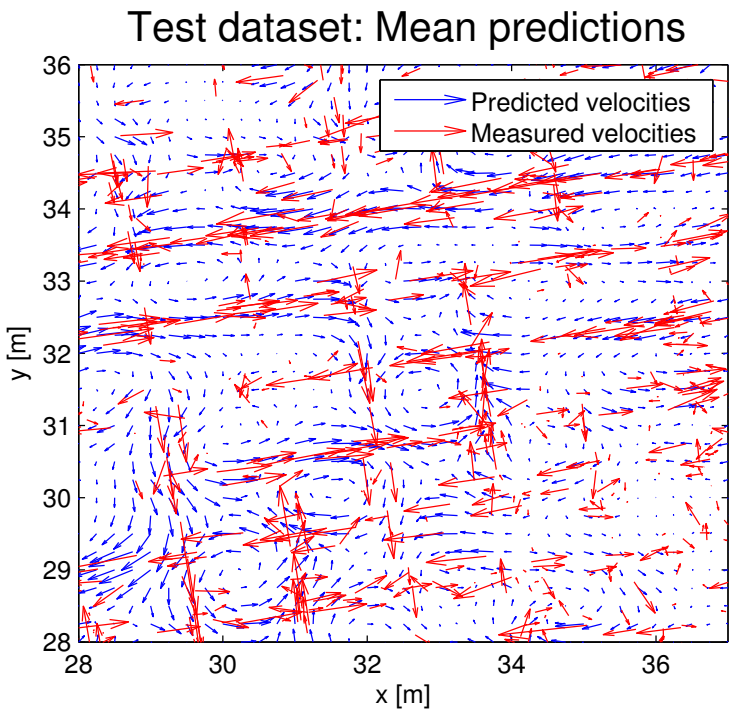
coordinates of taxis observed over a 24-hour subset of the month-long dataset. Figure 43 shows the predicted mean and variance values given by the GP regression using the learned hyperparameter values. The velocity measurements in the dataset are shown by red arrows, whereas the predicted velocities are shown in blue.

We use an empirical approach to learn the target survival and birth processes. For both processes, we overlay a uniform grid (1 m resolution) over the environment. Whenever a target appears in a cell, we add one to the survival count if the target was previously in another cell, we add one to the birth count if the target was previously outside the environment, and add one to the death count if at the next time step the target leaves the environment. The birth count for each cell is initialized to 10, so that the distribution of birth locations is uniform if there is no data. Similarly, the survival and death count for each cell are initialized to 9 and 1, respectively. The survival probability in a cell is given by the ratio of the survival count to the total survival and death counts in that cell. In the absence of data, this yields a uniform probability of survival of 0.9.

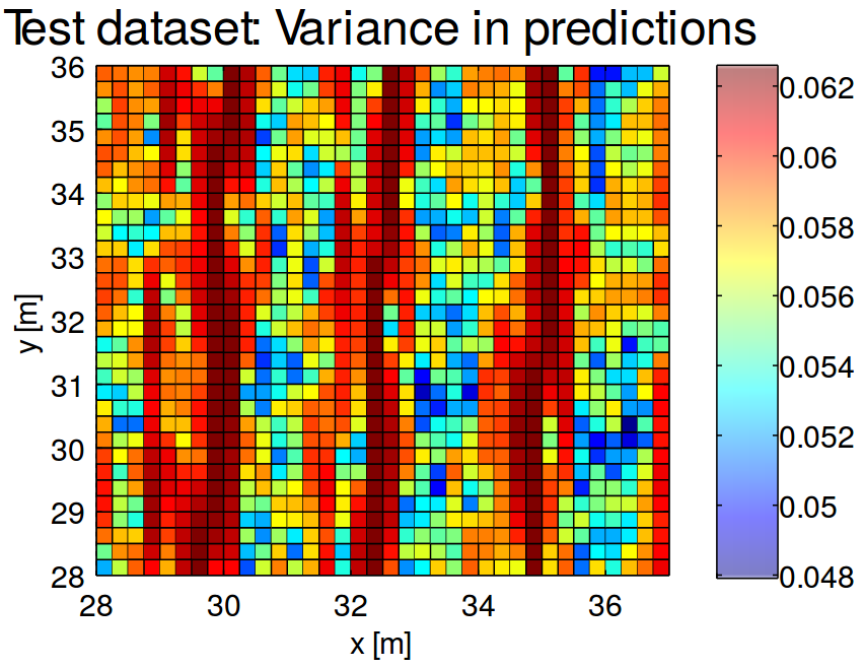
Figure 44a shows the environments used in the simulations, with the target survival probability in Figure 44b and birth PHD in Figure 45. As Figure 44b shows, the targets survive with high probability in the majority of the environment. The probability decreases near the western and southern edges of the environment, where there are roads along the edge of the environment. These same areas also have the highest rates of target births, as Figure 45b shows. One may also clearly see the highways in the southeast and the bridge in the northeast, which have the highest rates of traffic, and thus of target births and deaths. The target birth rate per minute, when considering all 536 taxis in the dataset, is 4.548 targets per minute of real time. The actual and fit birth rates are shown in Figure 45a, with the Poisson approximation fitting the data extremely well.

5.3.3 PHD Filter

We utilize the Sequential Monte Carlo (SMC) PHD filter from Vo et al. [102]. This approximates the PHD using a set of weighted particles, $v(x) \approx \sum_{i=1}^{P_t} w_i \, \delta(\mathbf{x} - \mathbf{x}_i)$. The SMC PHD filter allows for arbitrary, non-linear sensor and motion models, including a finite field of

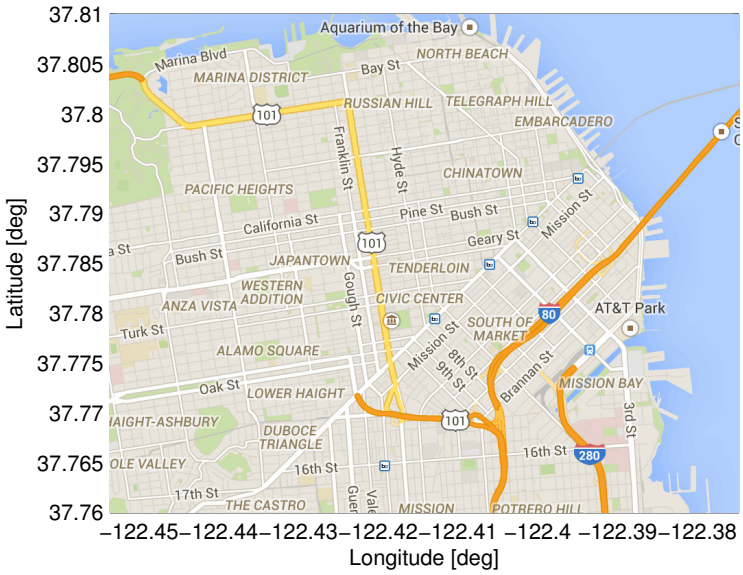


(a) Gaussian process mean

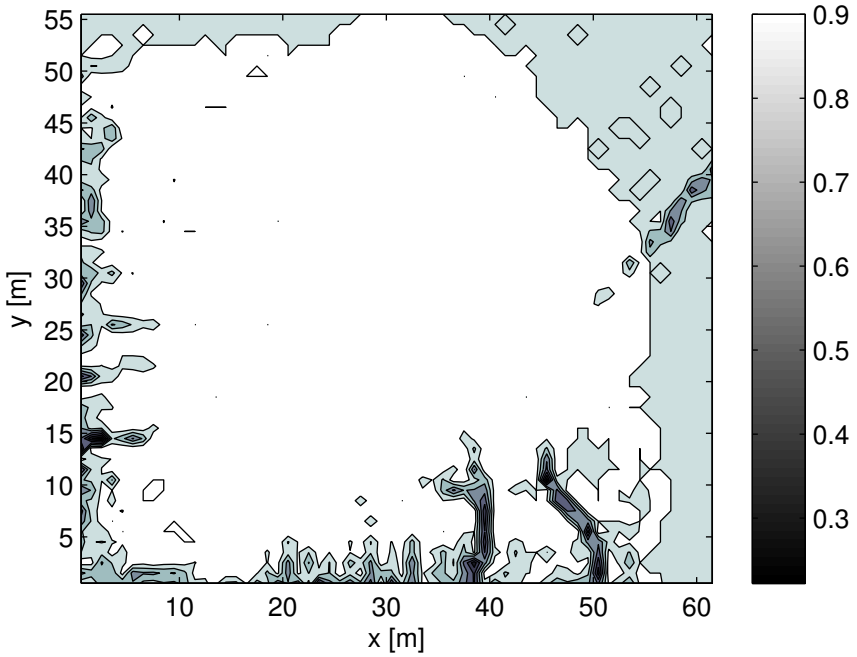


(b) Gaussian process covariance

Figure 43: The mean and covariance of the Gaussian Process regression motion model over a patch of the environment. The measured velocity vectors are shown in red, and the velocity vectors predicted over a grid are given in blue.

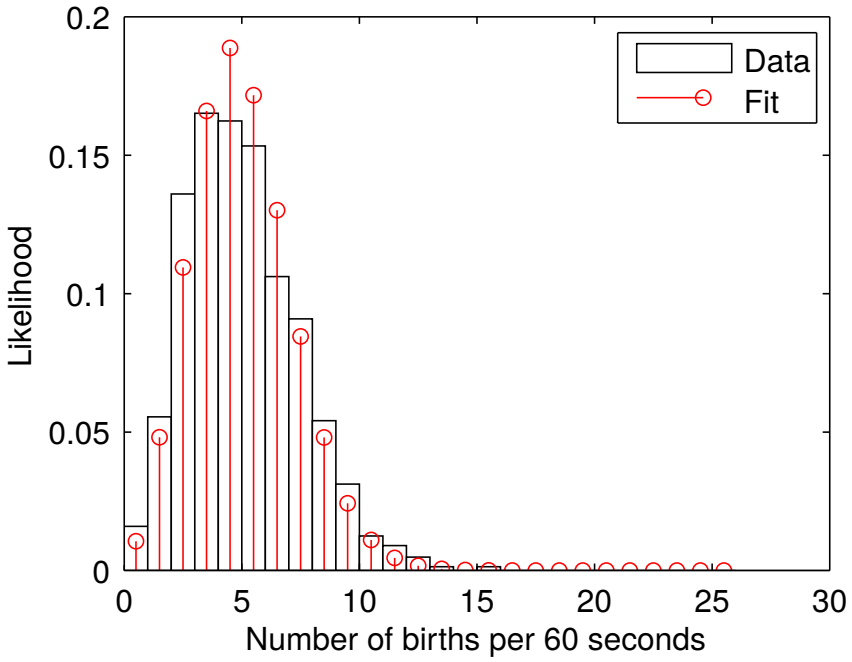


(a) Environment

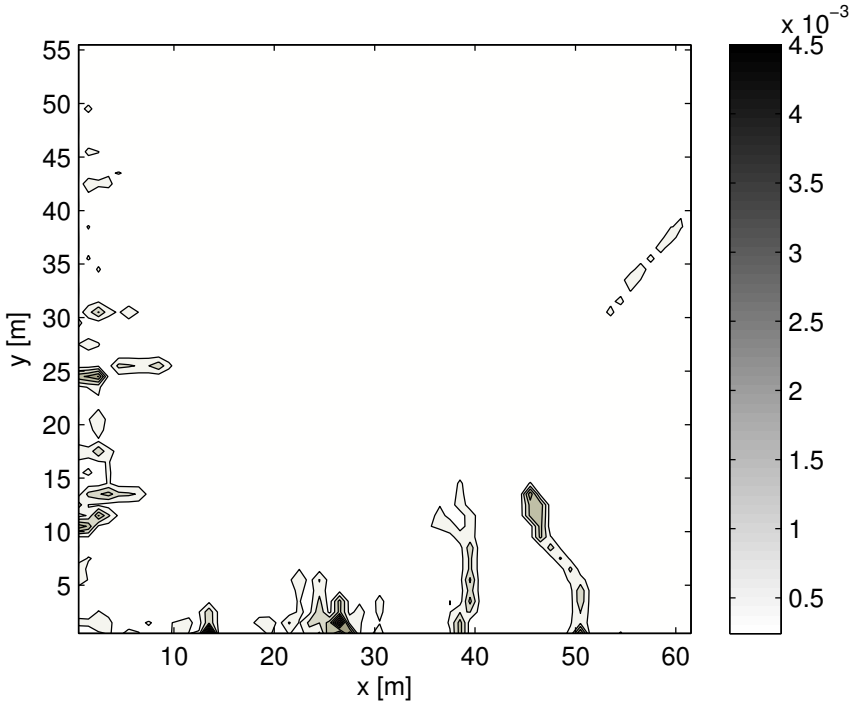


(b) Survival probability

Figure 44: (a) The area of interest, a roughly $6.15 \times 5.56\,\mathrm{km}$ region surrounding downtown San Francisco. (b) The probability of target survival as a function of position.



(a) Birth rate



(b) Birth PDF

Figure 45: Empirical target birth PHD.

view for the sensor. New particles are added to the PHD using the birth PHD described above as well as using the most recent measurement set and inverse measurement model, similar to the idea of Ristic et al. [86]. A fixed number of particles, P_b, are drawn from the birth PHD and an additional P_m particles are drawn from the inverse measurement model for each measurements in the most recent set, Z_t. The weight of each of these particles is $w = \frac{\int c(\mathbf{z})\, d\mathbf{z}}{P_b + |Z_t| P_m}$, where $|Z_t|$ is the cardinality of the measurement set. We utilize the low-variance resampling technique from Thrun et al. [97, Chapter 4].

5.3.4 Control Policy

In this section, we present our control policy for assigning trajectories for the robots. We study two objective functions for the control policy.

Mutual Information (MI) Objective

The robots utilize a receding horizon control policy similar to that from Chapter 4. Each robot generates a set of candidate trajectories, with T measurements along each trajectory at evenly spaced intervals. The optimal strategy is then to choose robot trajectories that maximize the mutual information between the target set and its future measurements,

$$Q_\tau^* = \underset{Q_\tau \in Q_\tau^{1:R}}{\operatorname{argmax}} I[\mathcal{X}_{t+T}; \mathcal{Y}_\tau^{1:R} \mid Q_\tau], \tag{5.4}$$

where $\tau = \{t+1, \ldots, t+T\}$ is the time horizon, $\mathcal{X}_{t+T}$ is the predicted location of the targets at time $t+T$, $\mathcal{Y}_\tau^{1:R}$ is the collection of binary measurements for robots 1 to R from time steps $t+1$ to $t+T$, and Q_τ are the future poses of the robots. These measurements depend on the future locations of the robots $Q_\tau = \{\mathbf{q}_{t+1}^1, \ldots, \mathbf{q}_{t+T}^1, \ldots, \mathbf{q}_{t+T}^R\}$. Computing Q_τ^* is computationally challenging, nevertheless, we show that a greedy strategy approximates Q_τ^* by a factor of 2.

We utilize binary measurements, rather than the full measurements sets, in order to decrease the computational complexity of the control policy. This allows us to derive a closed-form expression for (4.5), and we have previously shown that this approach effectively drives a team of robots to detect and localize static targets [25]. The binary measurements

are defined to be

$$y = \mathbf{1}\left(Z \neq \varnothing\right), \tag{5.5}$$

where $\mathbf{1}\left(\cdot\right)$ is the indicator function. Here $y = 0$ is the event that the robot receives no measurements to any (true or clutter) objects while $y = 1$ is the complement of this, *i.e.*, the robot receives at least one measurement. Kreucher et al. [60] take a similar approach, using a binary sensor model and an information-based objective function to schedule sensors to track an unknown number of targets.

Theorem 4. *The mutual information between the target set and the binary measurement model is a lower bound on the mutual information between the target set and the full measurement set,* i.e., $I[\mathcal{X}; \mathcal{Y} \mid Q] \leq I[\mathcal{X}; \mathcal{Z} \mid Q]$.

Proof. Note that y is deterministically related to Z, $y = \mathbf{1}\left(Z \neq \varnothing\right)$. This allows us to apply the Data Processing Inequality [20, Theorem 2.8.1], which states that functions of the data cannot increase the amount of information. □

We utilize a greedy approximation strategy to evaluate (4.5), similar to that used by Tokekar et al. [99]. Using this approach, each robot computes the utility of each action according to (5.4). The robot and action with the highest utility are selected. The remainder of the team then plans again, conditioned on the action of the first robot, and the robot and action with the highest utility are again selected. This process repeats until all robots have been assigned an action. Using the fact that mutual information is a submodular set function of robot poses, we can show that this greedy assignment policy is a 2-approximation.

Lemma 1. $I[\mathcal{X}; \mathcal{Y} \mid Q]$ *is a submodular set function of* Q.

Proof. See [57, Proposition 2]. □

Theorem 5. *Let* Q^G *be the robot poses selected by the greedy assignment policy and* Q^* *be the robot actions selected by the full, joint evaluation of* (4.5). *Then greedy is a 2-approximation,* i.e., $I[\mathcal{X}; \mathcal{Y} \mid Q^G] \geq \frac{1}{2} I[\mathcal{X}; \mathcal{Y} \mid Q^*]$.

Proof. We create a set system where the ground set, $\mathcal{X}$, is the target state. For each robot, let $\mathcal{Y}_i$ be the candidate measurement sets (induced by the candidate robot poses). The collection of $\mathcal{Y}_i$, say $\mathcal{Y}' = \{\mathcal{Y}_1, \ldots, \mathcal{Y}_N\}$, defines a partition matroid [7]: $(\mathcal{Y}', I)$ where any $\mathcal{Y} \subset \mathcal{Y}' \in I$ if and only if $|\mathcal{Y} \cap \mathcal{Y}_i| \leq 1$, for all i. That is, $\mathcal{Y}$ is a valid assignment of trajectories if and only if it chooses at most one trajectory corresponding to each robot. Mutual information is a submodular function as given by the previous lemma. □

Expected Number of Detections (END) Objective

The Expected Number of Detections (END) objective function is given by

$$N[X \mid Q] = \int \left(1 - \prod_{q \in Q} \left(1 - p_d(\mathbf{x} \mid \mathbf{q})\right)\right) v(\mathbf{x})\, d\mathbf{x}. \tag{5.6}$$

This objective gives the expected number of targets detected by at least one robot, and is a submodular set function of Q so the greedy assignment algorithm will be a 2-approximation, similar to the previous theorem.

Lemma 2. *The END objective function, $N[X \mid Q]$, is a submodular function of Q.*

Proof. The difference in the objective when adding a single robot is

$$N[X; Q \cup \{\mathbf{q}'\}] - N[X; Q] = \int p_d(\mathbf{x} \mid \mathbf{q}') \prod_{\mathbf{q} \in Q} \left(1 - p_d(\mathbf{x} \mid \mathbf{q}) \bigcap\right) v(\mathbf{x})\, d\mathbf{x}.$$

For any $R \subseteq Q$, the product $\prod_{\mathbf{r} \in R} \bigcap (1 - p_d(\mathbf{x} \mid \mathbf{r})) \geq \prod_{\mathbf{q} \in Q} \left(1 - p_d(\mathbf{x} \mid \mathbf{q})\right)$ since $p_d(\mathbf{x} \mid \mathbf{q}) \in [0, 1]$ $\forall \mathbf{x}, \mathbf{q}$. Thus $N[X; R \cup \{\mathbf{q}'\}] - N[X; R] \geq N[X; Q \cup \{\mathbf{q}'\}] - N[X; Q]$, so by definition $N[X, Q]$ is submodular. □

Trajectory Generation

We use a simple model for a fixed-wing aircraft with three basic control inputs: forward velocity, yaw rate, and pitch rate. For each control input we select a range of possible values. For each possible set of control inputs we integrate the position, yaw, and pitch forward in time using a 1-step Euler integration scheme. Any trajectories that bring the

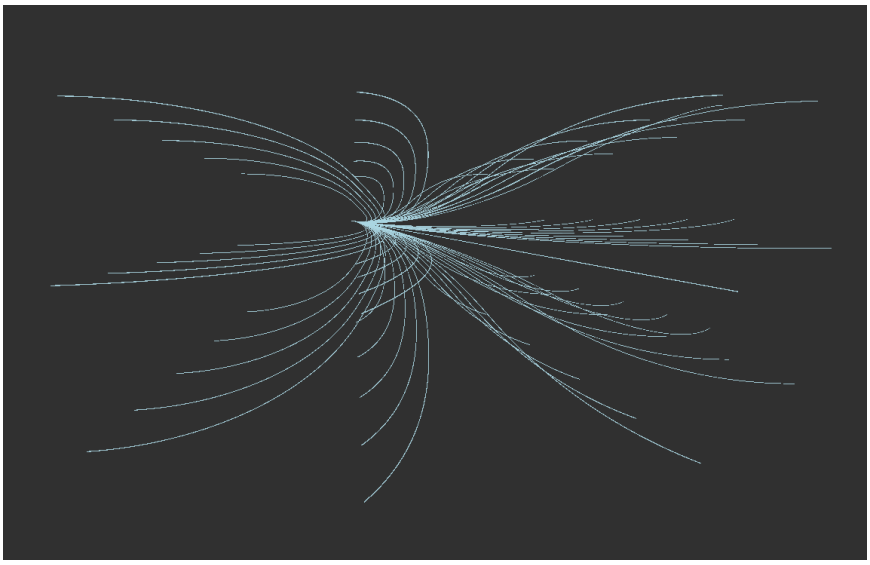


Figure 46: Sample trajectories.

robots above or below the elevation limits are discarded as invalid, as are any that result in collision. The remaining trajectories are interpolated to yield the T poses at which each robot will take a measurement. Figure 46 shows sample trajectories for a single robot.

5.4 Results

To test the performance of the different objective functions, we ran a series of simulated experiments varying the number of robots, the length of the planning horizon, the objective function, and the target motion model. We used teams of 2, 4, and 6 robots, either keeping the number of planning steps constant ($T = 2$) or keeping the total number of actions for the team constant ($RT = 12$). We perform 5 trials with each configuration, using a random subset of 80 targets from the taxi database for the ground truth target motion. Note that the true number of targets in the area of interest varies over time as targets enter and leave. The robots monitor the area from Figure 44a, which is scaled down by a factor of 100. We also sped up the data by a factor of 60, so 1 s in simulation represents 1 min of real time, in order to speed up the simulations. The data is taken from 5–9 pm on May 18, 2008, a time of day where there will be plenty of taxi traffic.

5.4.1 Moving Targets

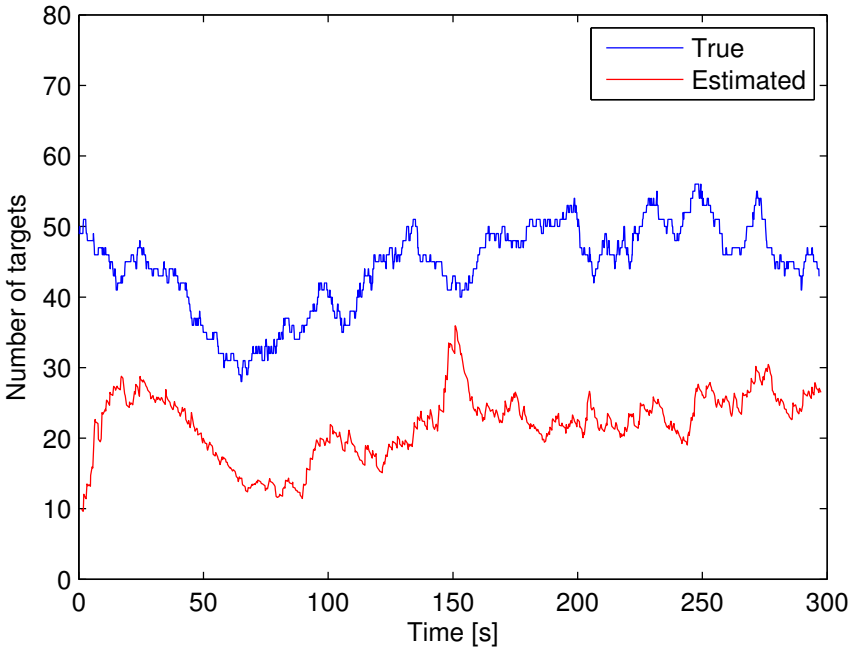
The two target motion models that we consider are the Gaussian process (GP) described in Section 5.3.2, and a Gaussian random walk (GRW) model. In GRW we model the target as performing a random walk, with a velocity drawn at random from a Gaussian distribution. Note that these models are used only to update the PHD; the actual targets trajectories are given by the taxi dataset. In both cases we use the survival and birth processes described in Section 5.3.2, with the birth rate set to 0.6788 to account for the reduced number of data files used.

Figures 47–49 show how the ratio of the expected number of targets to true targets, the robot elevation, and the target set entropy change during a single run. These are representative trials of a team of 2 robots with a planning horizon of 6 time steps.

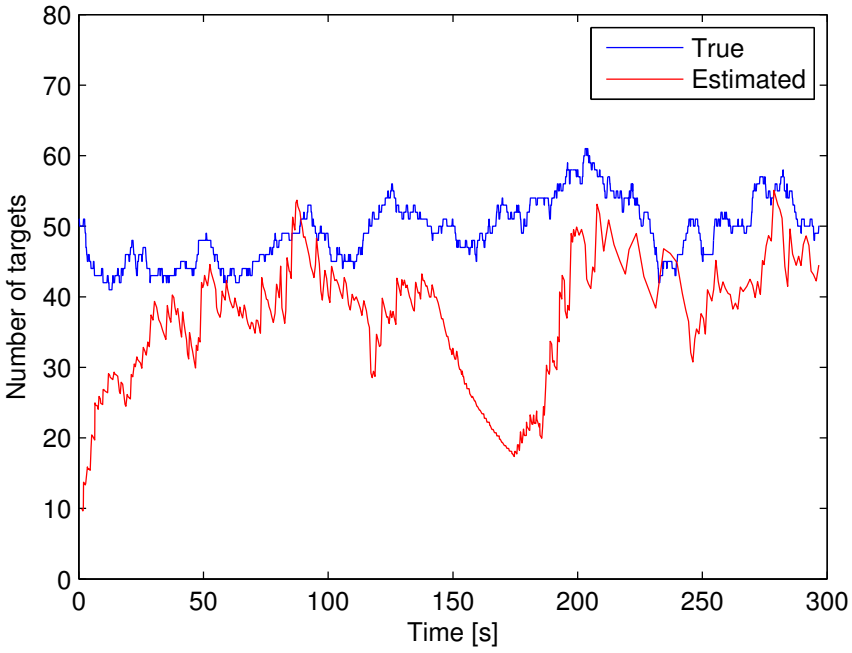
Figures 50–52 show the ratio of the expected number of targets to true number of targets, the fraction of true targets within the sensor FoV, and the ratio of expected targets to true targets within the sensor FoV, respectively. In general, the fraction of targets tracked by the team depends much more on the motion model and the objective function than on the team size or planning horizon, despite the fact that larger teams and planning horizons cause the robots to observe a larger number of targets. Additionally, the ratio of the expected number of targets to the true number of targets within the sensor FoV is largely independent of the objective function, team size, or planning horizon.

Overall, the robot teams using the information based control objective (MI) estimate and track fewer targets than the teams using the END objective but each target is tracked with higher quality. Additionally, the teams using the GP-based motion model track more targets than those using the GRW motion model.

The reason for this difference is due to the emergent behavior of the different control objectives. Robots using the MI objective tend to stay closer to the ground in order to decrease uncertainty in the location of individual targets. On the other hand, robots using the END objective fly at a higher altitude, as Figure 53 shows. Note that increasing the altitude decreases the probability of detection, while increasing the sensor FoV. Consequently,

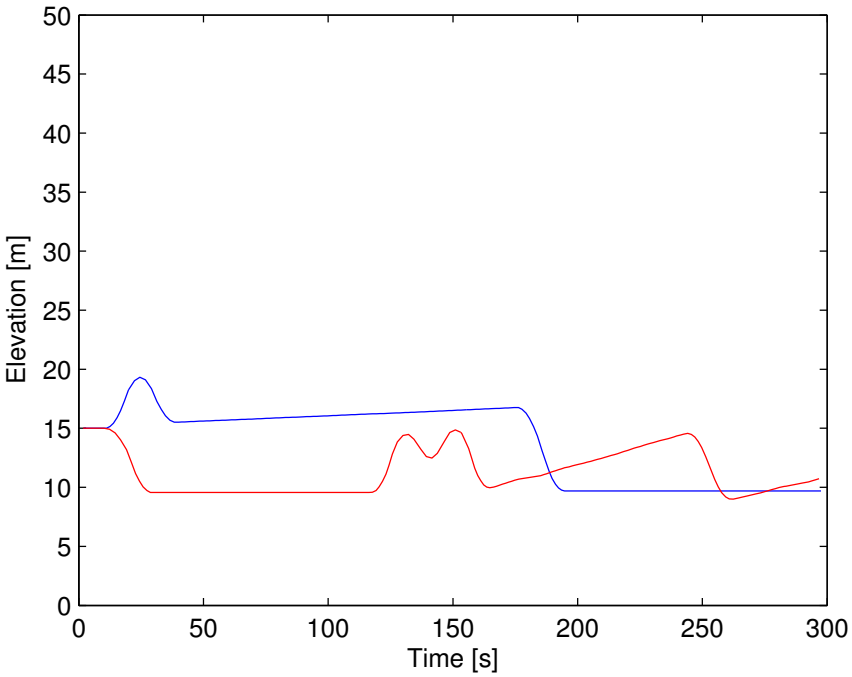


(a) GP motion model with MI objective

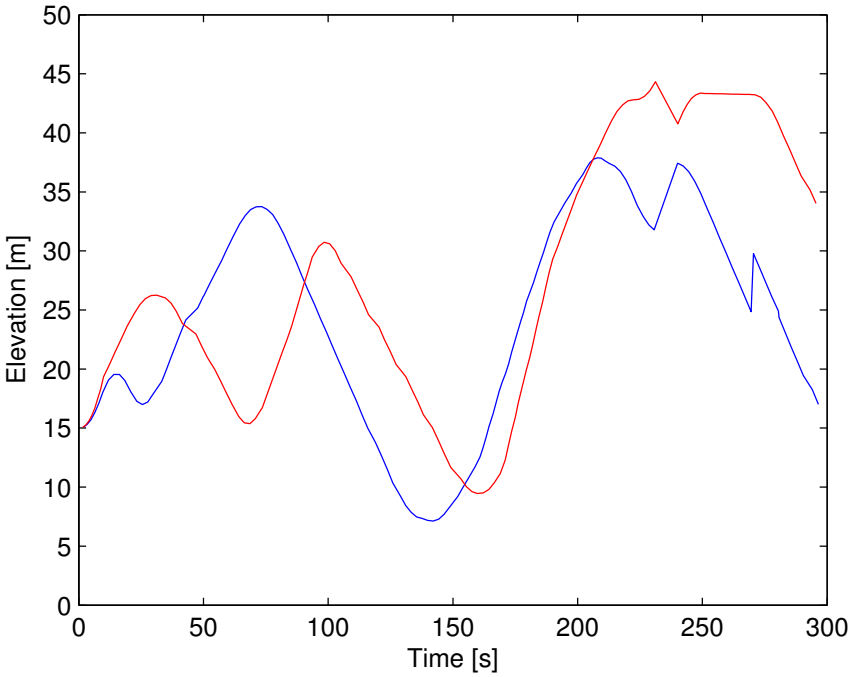


(b) GP motion model with END objective

Figure 47: Ratio of the expected number to the true number of targets over a single run for $R = 2$ and $T = 6$.

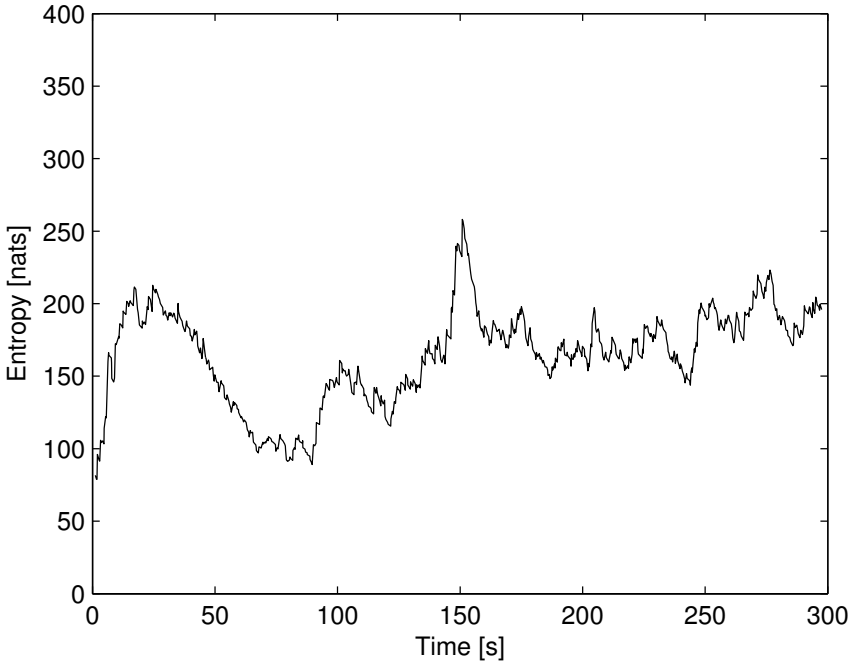


(a) GP motion model with MI objective

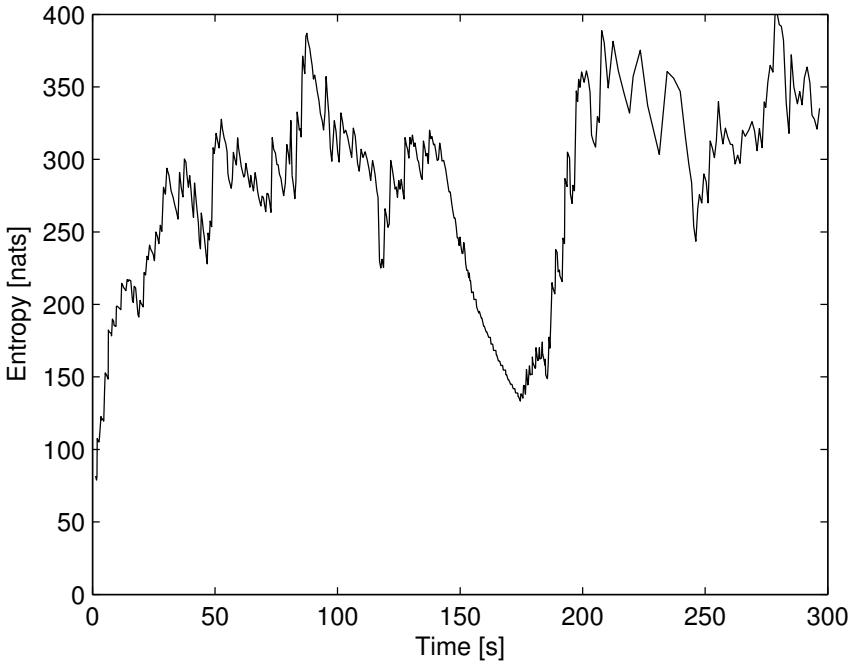


(b) GP motion model with END objective

Figure 48: The elevation of the robots over a single run for $R = 2$ and $T = 6$.



(a) GP motion model with MI objective



(b) GP motion model with END objective

Figure 49: The entropy of the target set over a single run for $R = 2$ and $T = 6$.

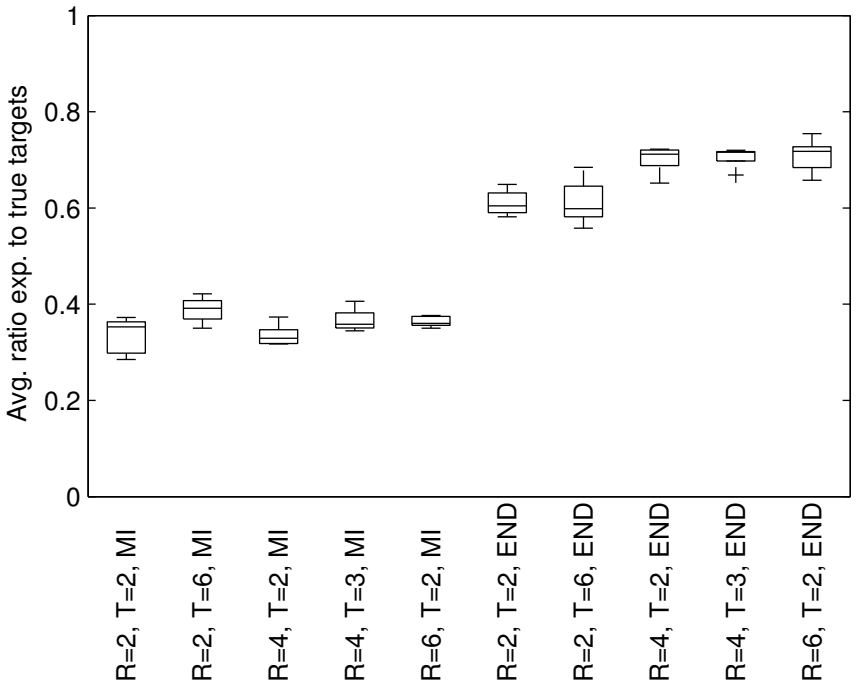
flying to the highest altitude is not necessarily optimal. Figure 54 shows the average target entropy. This is substantially lower for the teams using MI, indicating that the targets are being tracked with less uncertainty.

5.4.2 Static Targets

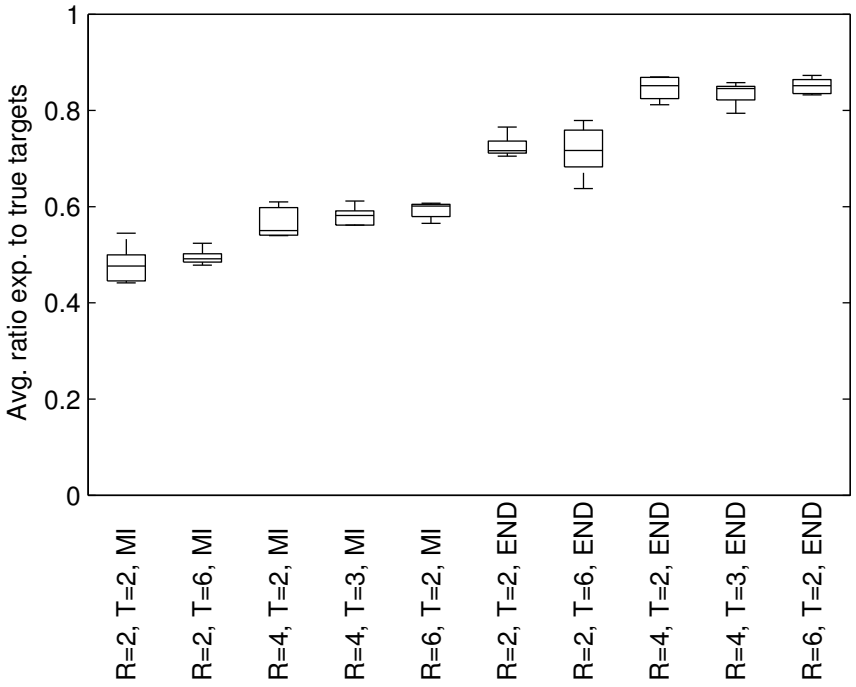
We also test the performance of our framework with static targets using a team of 4 robots with a planning horizon of 3. The simulation parameters are identical, except we replace the 80 taxi data traces with 80 randomly drawn static target locations. The resulting final estimated number of targets and target entropies are shown in Figure 55. The final estimated number of targets is very close to 1 using both objective functions, indicating that the system is able to correctly determine the number of targets. The entropy is also lower than in the case of moving targets.

5.5 Conclusion

In this chapter we describe a framework for detecting, localizing, and tracking an unknown number of moving targets using a team of mobile robots. The robot team uses the Probability Hypothesis Density filter to simultaneously estimate the number of targets and the states of the targets. The PHD filter is robust to false negative and false positive detections and sensor noise and does not require any explicit data association. Using the estimate of the target set from the PHD filter, the robots greedily select actions that maximize submodular control objectives. The two control objectives that we consider in this paper are the expected number of detected (END) targets by the team and the mutual information (MI) between the predicted targets and the future detections of the robots. We validate our framework through extensive simulations using a real-world dataset for target motion. Robot teams using the END objective track a higher fraction of the targets but do not localize the targets with high precision. Conversely, robot teams using MI track a smaller number of targets but have significantly lower uncertainty in the target positions.

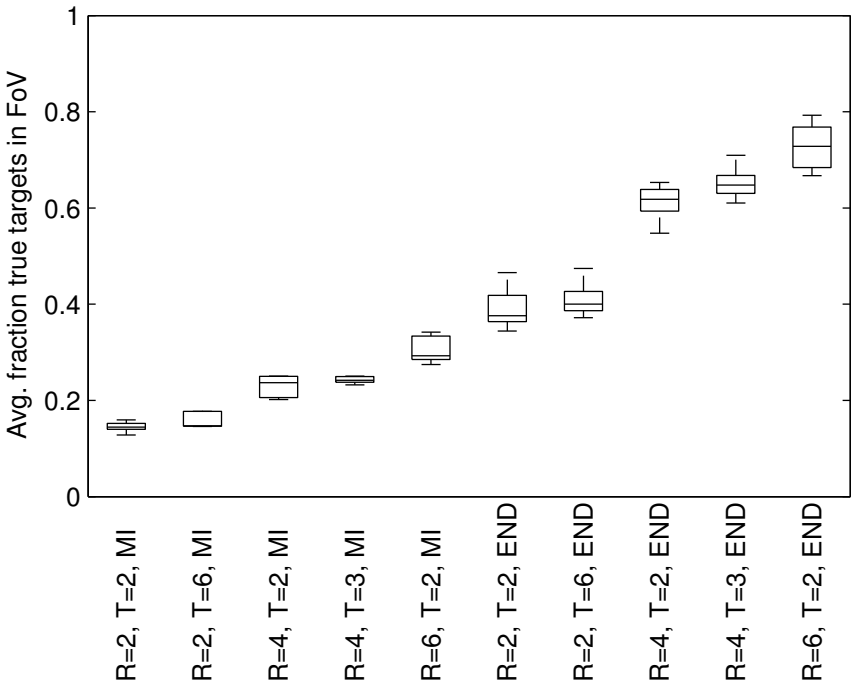


(a) Gaussian random walk

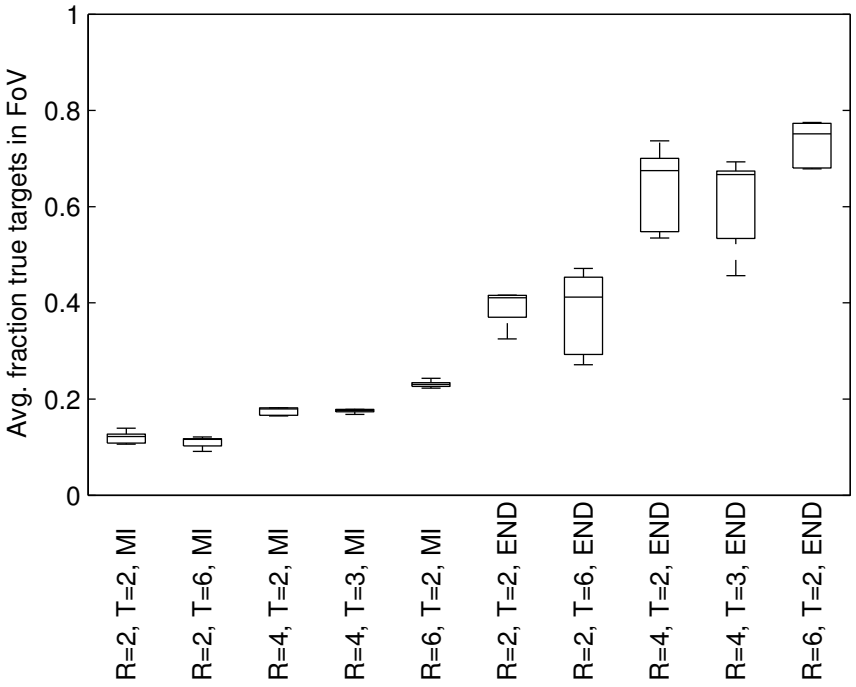


(b) Gaussian process

Figure 50: Average ratio of the expected number of targets to the true number of targets over a single run.

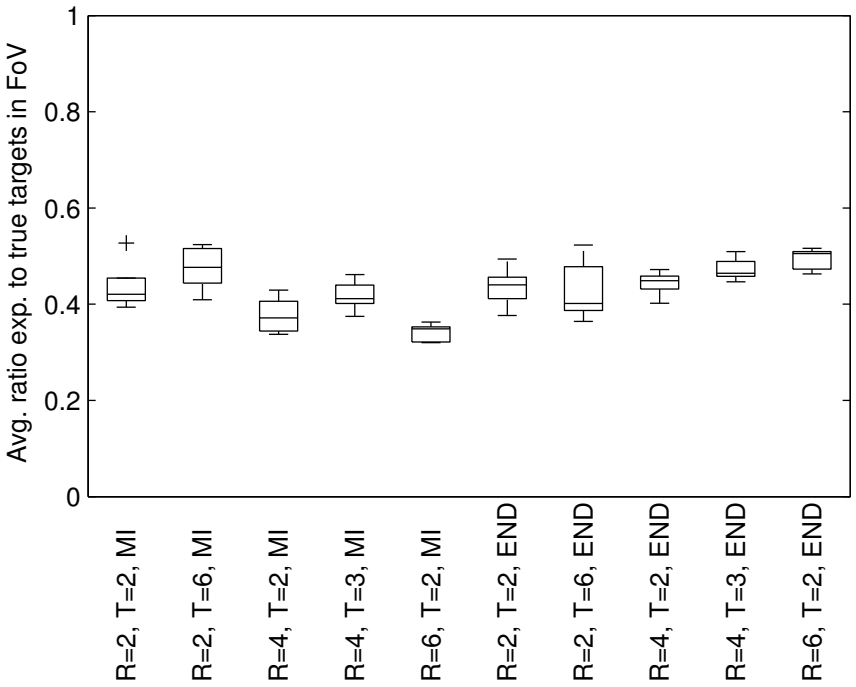


(a) Gaussian random walk

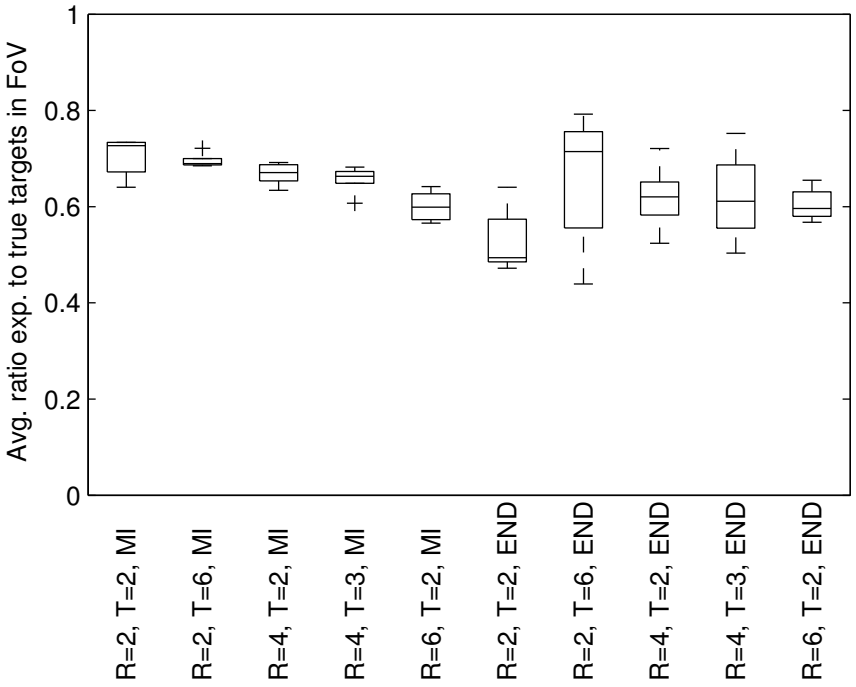


(b) Gaussian process

Figure 51: Average fraction of the number of true targets within the team's field of view over a single run.

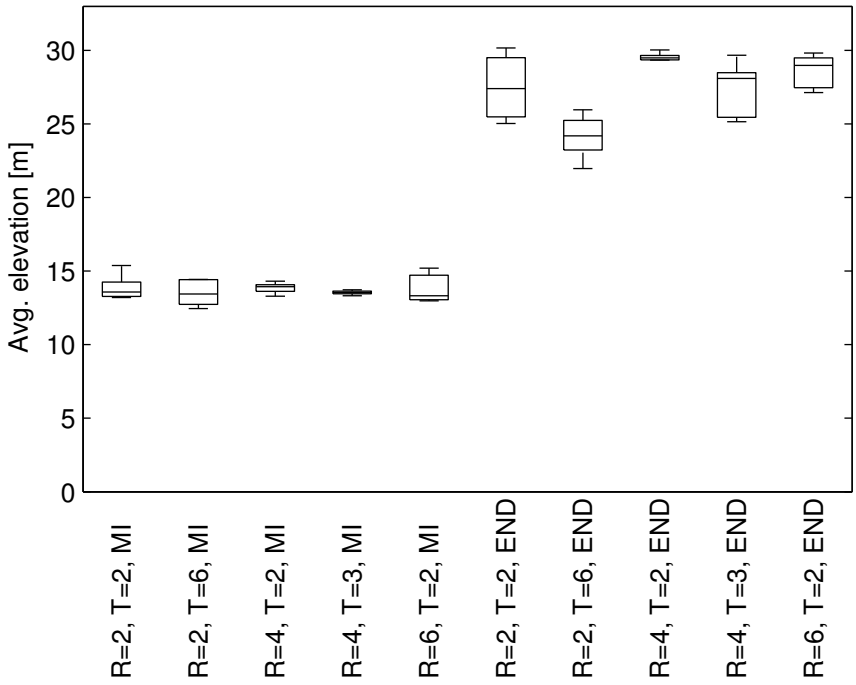


(a) Gaussian random walk

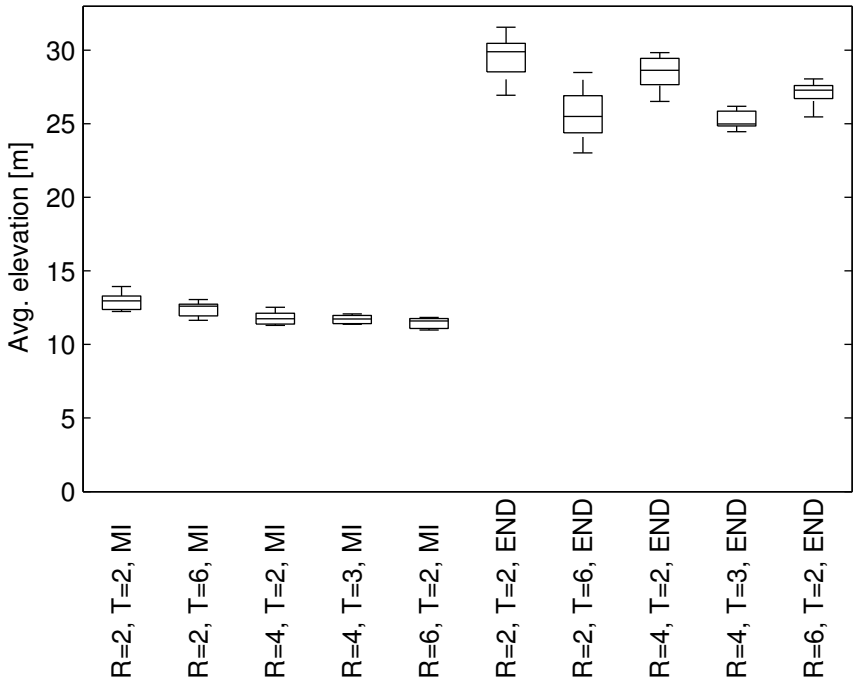


(b) Gaussian process

Figure 52: Average ratio of the expected number of targets to the true number of targets within the team's field of view over a single run.

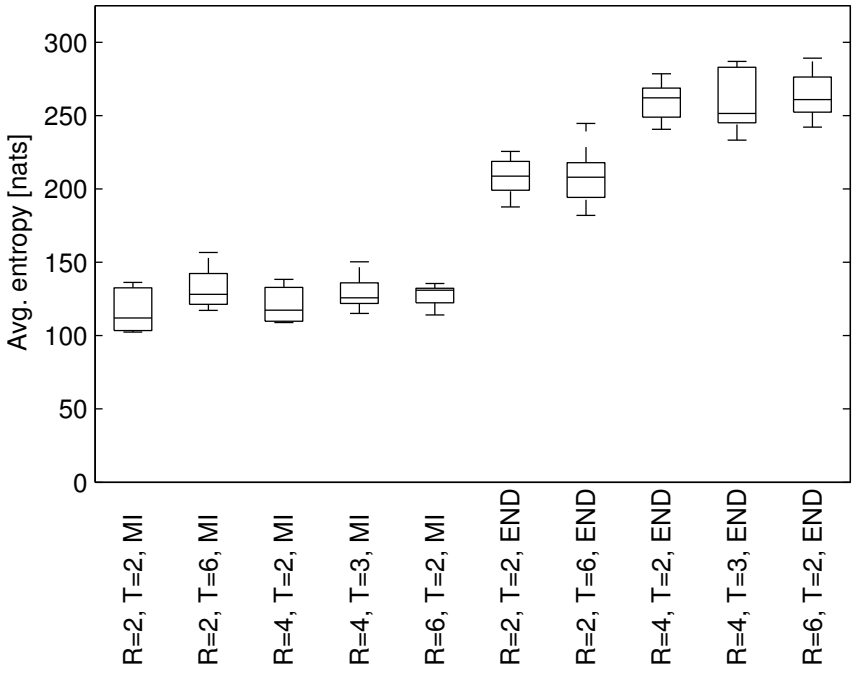


(a) Gaussian random walk

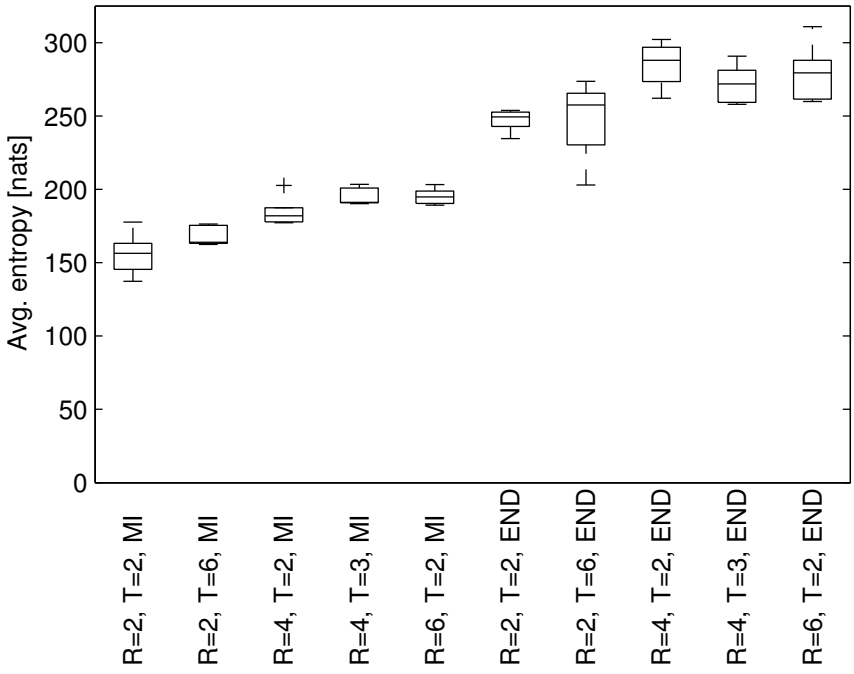


(b) Gaussian process

Figure 53: Average elevation of the robots over a single run.

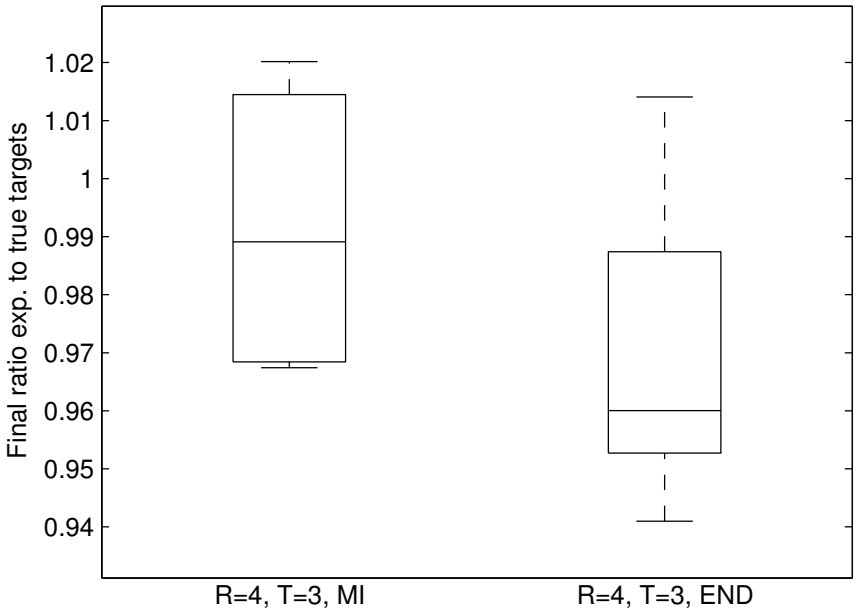


(a) Gaussian random walk

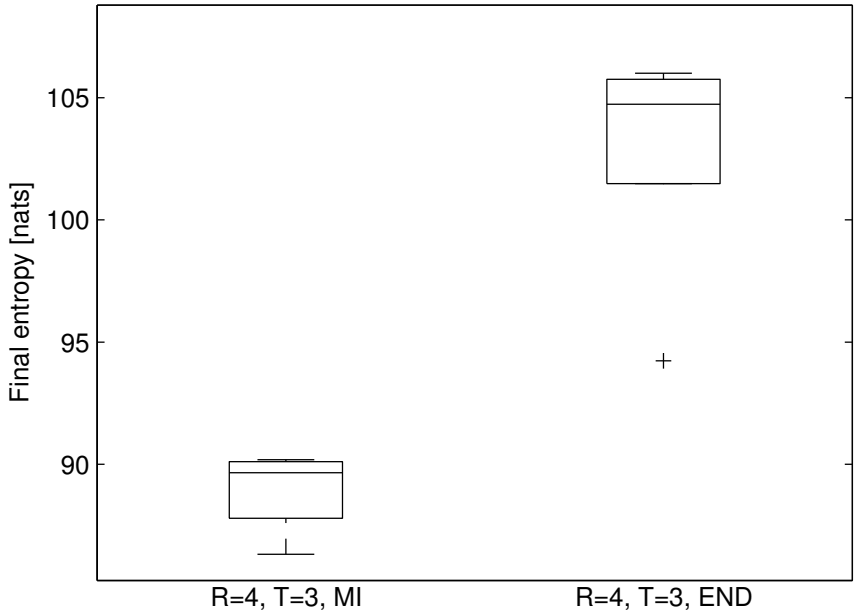


(b) Gaussian process

Figure 54: Average entropy of the target set over a single run.



(a) Final ratio of the expected number of targets to the true number of targets.



(b) Final entropy of the estimated target set.

Figure 55: Performance of our framework with static targets.

Chapter 6

Conclusion

6.1 Contributions

This dissertation presents an active information gathering framework for small teams of robots. The formulation, which is based on random finite sets, allows us to apply this framework to situations in which there is ambiguous or unknown data association and the number of objects of interest is not known at the beginning of the information gathering task. Such situations are commonplace in real-world applications, including security and surveillance, infrastructure inspection, environmental monitoring, precision agriculture, landmark localization, and map building. In some of these tasks, such as infrastructure inspection or security, the number of objects of interest is typically small and their locations may be correlated – if a robot detects damage to one section of a bridge then the nearby areas may also be damaged or weakened, or the motion of two intruders in a restricted area may be coordinated. In other tasks, such as environmental monitoring or map building, the number of features may be very large and there may be multiple objects of interest that the team must detect and localize.

Our information gathering framework functions well in any of these scenarios. Chapter 2 presents background material on multi-target tracking, finite set statistics, and information-based control, the three core concepts of our information gathering framework. Chapter 3 presents an estimation algorithm based on random finite sets that tracks a small number of

objects using a team of robots. The robot team follows the gradient of mutual information, an inherently local approach. This behavior is useful in security and surveillance settings, where we would like a robot team to track intruders. We demonstrate the utility of this approach through simulation and hardware experiments with the CANINE robot and with a small team of quadrotor MAV platforms.

In Chapter 4 we use the PHD filter to track a large number of targets, and the robot team considers actions over a range of length scales. This allows the team to decide online whether it is more useful to take repeated measurements of a local region or to explore more distant regions. The team executes the plan over a receding horizon, adapting to the information contained in the incoming measurements. The team may also interact with a central information server and compare the relative information benefits of locally sensing the environment versus communicating with the network resources. We demonstrate the performance of this approach through a series of simulation and hardware experiments with a team of Scarab robots.

Chapter 5 extends this approach to the case of moving targets. We learn target motion models using Gaussian Process regression on a real-world dataset containing the motion of over 500 taxis over the course of a month. We present experiments with a small simulated team of fixed-wing aircraft tracking 80 moving targets over a simulated area that is tens of square kilometers.

6.2 Future Work

6.2.1 Risk Avoidance

In Chapter 3, the robot team was able to discover regions that could be hazardous and took the likelihood of failure into account in the control objective. This discovery was done by detecting robot failure, an expensive method of "sensing" such hazards. Being able to actively detect these hazards would allow robots to generate plans and select actions that keep them safe while operating in hazardous environments, *e.g.*, in search and rescue or first responder scenarios. Users could also specify the desired level of risk that they are willing

to tolerate for a particular application. In certain time-critical situations, the users may be willing to risk losing some individual agents in the hope of achieving the goal more quickly. This risk could also be automatically tuned online, becoming more conservative if agents fail, to ensure that some agents will survive in order to complete the mission. Operating in these potentially hazardous environments may also require different mobility and sensing capabilities of the robots. For example, there may be damaged buildings or infrastructure that require a combination of ground and aerial robots to fully explore. Robots may also need to seek out radiation sources, gas leaks, or fires in addition to trapped or injured people, requiring multiple sensing modalities.

6.2.2 Active SLAM

The approaches presented in this dissertation all assume that the robots are able to localize themselves with high accuracy and that the search area is known *a priori*. It would be beneficial to extend this work to situations with unknown maps and uncertain localization. For example, in a search and rescue scenario where a team of robots is searching for victims in a collapsed building, even having the building blueprints available will not be sufficient for the robots to localize themselves since the building may have experienced significant damage. In such settings it is important for a robot team to be able to build an accurate map online as it explores and to localize itself within such a map, a problem known as simultaneous localization and mapping (SLAM). SLAM has been an area of active research within the robotics community for decades and there are a number of existing solutions to the problem of active SLAM. Carrillo et al. [10] present simulations and experiments utilizing a new utility function for active SLAM using an occupancy grid representation [32] of the environment. Mullane et al. [76, 77] provide the first solution to the SLAM problem based on FISST, looking at the problem of landmark-based mapping. They utilize the Rao-Blackwellized particle filter and represent the landmarks using an RFS, using the PHD filter, or some variant of it, to recursively update the estimate of the map landmark set conditioned on the trajectory of the robot. They show that this approach leads to superior performance compared to traditional approaches to SLAM that utilize heuristic methods for

data association and map management. Moratuwage et al. [75] extend the RB-PHD-SLAM algorithm to use multiple feature classes and to work with multiple robots. Each feature class has a different motion model associated with it, allowing the robot team to track static and mobile targets. Extending active information gathering approaches to the problem of active SLAM, using the RB-PHD-SLAM algorithm, would allow robots to better reason about sensor and environmental uncertainties while autonomously exploring new environments.

6.2.3 Extension to Other Estimation Algorithms

The FISST community has recently developed new estimation algorithms based on random finite sets. The Cardinalized Multi-target Multi-Bernoulli (MeMBer) filter [104] and labeled MeMBer filter [84] represent targets using the so-called Bernoulli RFSs. A Bernoulli RFS has two components, a probability of existence and a probability density of the state of the target, and represents the estimate of a single target. The MeMBer filter recursive updates a collection of independent Bernoulli RFSs, merging sets that represent the same target and pruning sets with low probabilities of existence. This approach, particularly the labeled version, makes it easier to extract estimates for individual targets. Extending the active information gathering framework to utilize these new filters would be beneficial, particular when tracking moving targets when it is more difficult to extract individual target tracks from a time history of the PHD. This will requires us to derive a new, computationally tractable expression for the mutual information between the target set and the future measurements of the robots.

6.2.4 Interacting With the Internet of Things

According to the McKinsey Global Institute, there are twelve key technologies that will transform the way we live [71]. Among those are advanced robotics, the Internet of Things (IoT), and cloud technology. As the IoT continues to expand, wireless devices and sensors will become increasingly prevalent in the environment around us. Robots will play a part in discovering and interacting with these devices, using their mobility to collect data from isolated sources to share with other devices or to upload information to the cloud. Addition-

ally, robots may utilize the cloud infrastructure to perform resource-intensive computations or to gain access to large-scale databases. Chapter 4 touches on this idea, allowing robots to upload and download data from a central server and considering the relative merits of collecting information via direct sensing versus via communication. This synergy between the IoT infrastructure and mobile robot platforms offers many new research opportunities in the discovery, collection, and dissemination of information. For example, a mobile robot could upload a video stream collected during exploring and receive a set of semantic object labels along with the image processing software and models necessary to detect future instances of these objects. Robots must also learn to utilize increasingly heterogeneous data. For example, in smart building or smart city applications, robots could use either onboard sensors or connect to dozens of different smart sensors embedded in the local environment to quickly gain information relevant to the current mission.

6.3 Concluding Remarks

This dissertation presents a unified estimation, control, and communication framework for multi-robot information gathering. The framework is applicable to a variety of different real-world tasks and can be applied to robots with different mobility and sensing capabilities. An extensive series of simulated and hardware experiments verify the performance of the framework. Future work will extend this framework to more explicitly consider exploration in hazardous environments and to consider uncertainty in robot localization. This will allow the framework to be used in a broader set of tasks, including first responder scenarios and search and rescue missions. We plan to improve the performance of the system by utilizing recent advances in multi-target estimation and tracking algorithms, allowing the robots to more quickly localize targets and to more easily extract target tracks. Finally, as robots continue to play an increasingly large role in daily life, they must become aware of and utilize the increasing number of available heterogeneous resources, from smart sensors embedded in the local environment to cloud computing. We hope that all of the work contained in this dissertation will help make life more productive and safe.

Appendices

Appendix A

Magnetic Anomaly Detection Sensor Characterization

We ran experiments to experimentally characterize the detection statistics for the magnetometer on board the AscTec Hummingbird MAV platform, shown in Figure 56. The magnetometer has 3 measurement axes and the magnitude of the vector appears to be approximately normalized to the strength of the Earth's magnetic field. The targets are cylindrical neodymium magnets[1] with the axis of the magnet aligned with the z-axis of the global coordinate frame.

To learn the baseline magnetic field, we flew the Hummingbird through a lawnmower pattern over a $2.6 \times 2.6\,\mathrm{m}$ area at a constant height of approximately 1.3 m. Figure 57 shows that the magnetic field experiences significant changes over the area covered by the robot, varying by almost an order of magnitude. This is likely due to the presence of a metal staircase and other large objects in addition to building materials, electrical wires, and other robots in the laboratory space. Despite the large variance in measurements taken at the same location, the average field changes smoothly over the environment. We fit a nominal field strength by dividing the area into grid cells with size 30 cm and taking the empirical mean of all of the magnetic field readings taken within that cell.

[1] K&J Magnetics, Inc. D8Y0, `https://www.kjmagnetics.com/proddetail.asp?prod=D8Y0a`

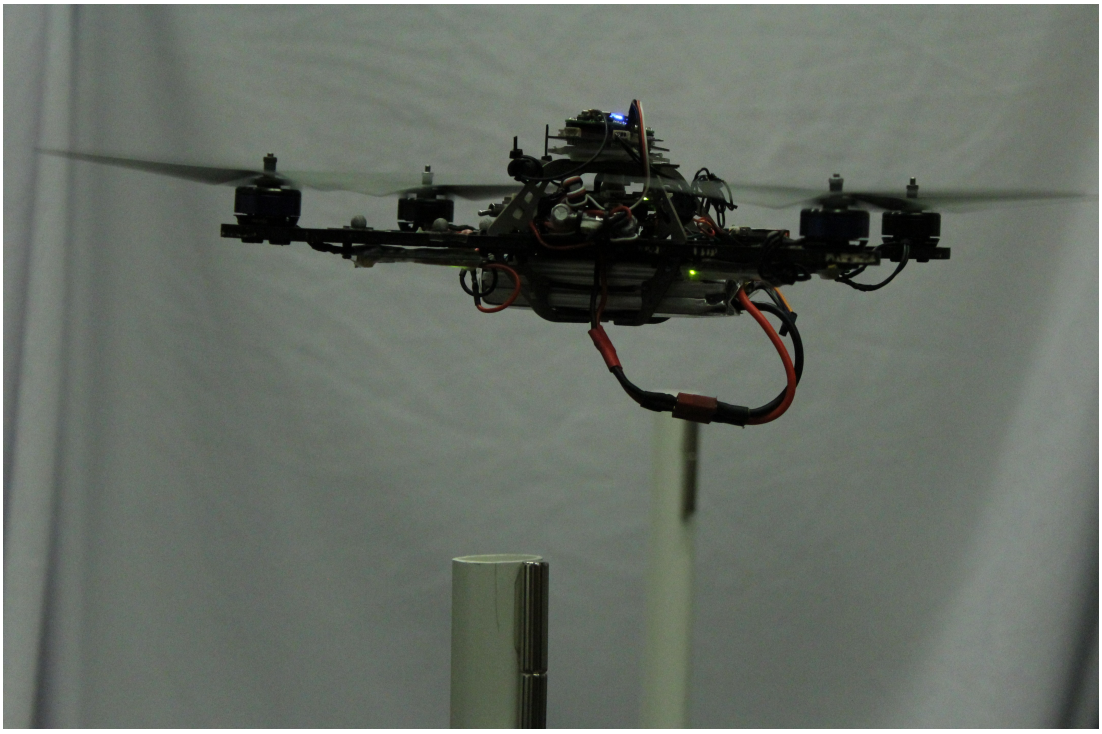
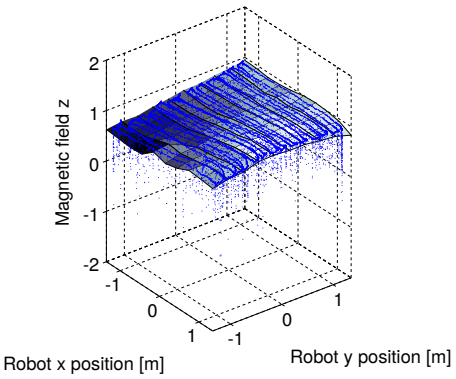


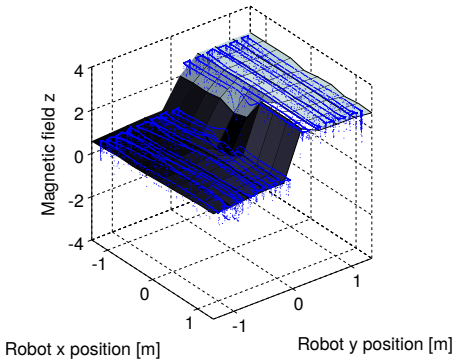
Figure 56: Photo of an Ascending Technology Hummingbird MAV hovering over a magnetic target. A second target may be seen in the background.

Note that for MAV Kilo there is a bias induced in the readings along the z-axis of the magnetometer after the robot passes directly over the magnet. This does not occur for MAV Papa. The phenomenon was repeatable in the training runs, but presents problems for actual experimental trials as the robot does not know the locations of the magnets *a priori*. To avoid this problem, we instead use magnetometer along the x-axis for MAV Kilo. However, the deviation along the x-axis, unlike the z-axis, is not isotropic and changes signs depending on what side of the magnet the robot is on, as Figure 57d shows. In order to keep the models for the two robots similar, we ignore this change in the sign of the deviation for MAV Kilo and only consider the absolute value of the deviation in the magnetic field.

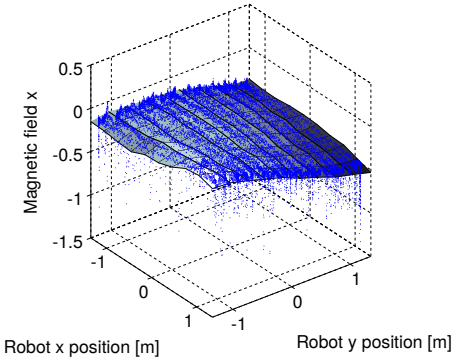
In order to determine the presence of a target, we use the deviation from the nominal field shown in Figure 57. To characterize the detection statistics, we flew two more lawn-mower patterns with a single magnetic target positioned at $(0, 0, 1.12)^T$ m. The resulting deviations in the magnetic field are plotted as a function of the distance to the target in



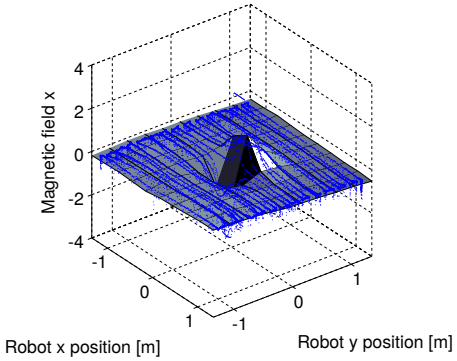
(a) MAV Kilo – z-axis – no magnet.



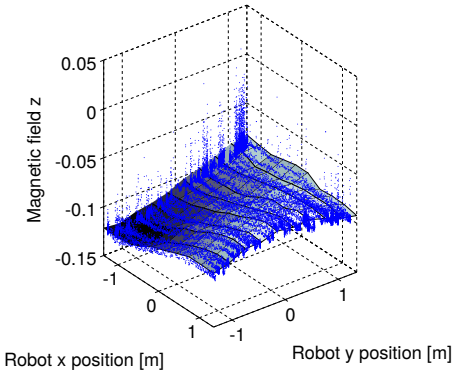
(b) MAV Kilo – z-axis – magnet.



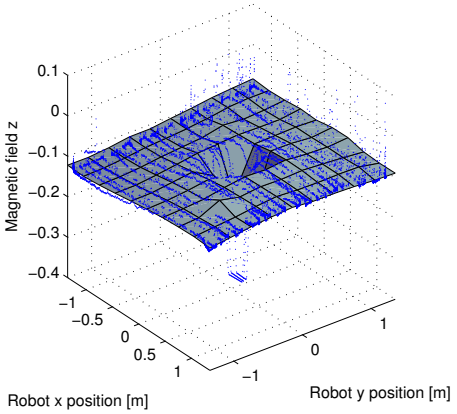
(c) MAV Kilo – x-axis – no magnet.



(d) MAV Kilo – x-axis – magnet.

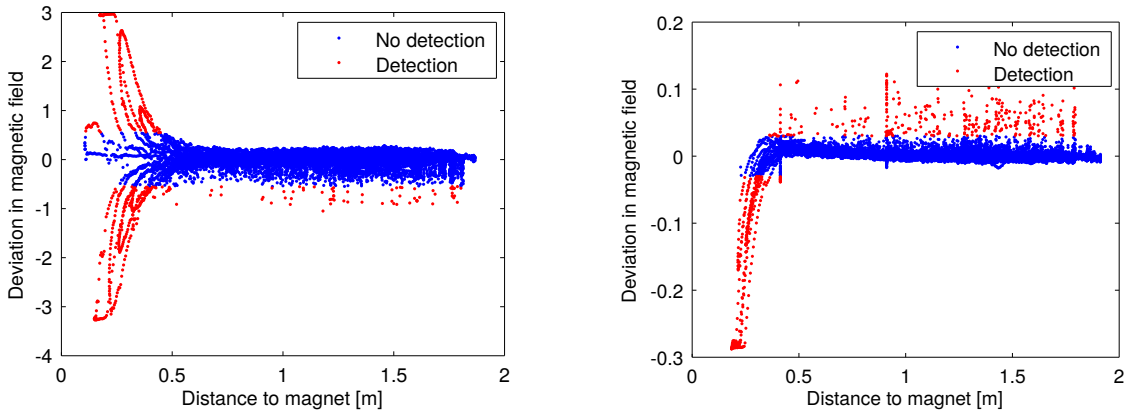


(e) MAV Papa – z-axis – no magnet.



(f) MAV Papa – z-axis – magnet.

Figure 57: Experimental results of the magnetic field strength as a function of the 2D position of the MAVs in the baseline training runs. Blue dots indicate the individual data points and the grayscale surface shows the average value in each cell. The magnetic field strength along the z-axis for MAV Kilo is shown (a) before and (b) after a magnet is placed at (0, 0) m. The magnetic field strength along the x-axis for MAV Kilo is shown (c) before and (d) after a magnet is placed at (0, 0) m. The magnetic field strength along the z-axis for MAV Papa is shown (e) before and (f) after a magnet is placed at (0, 0) m.

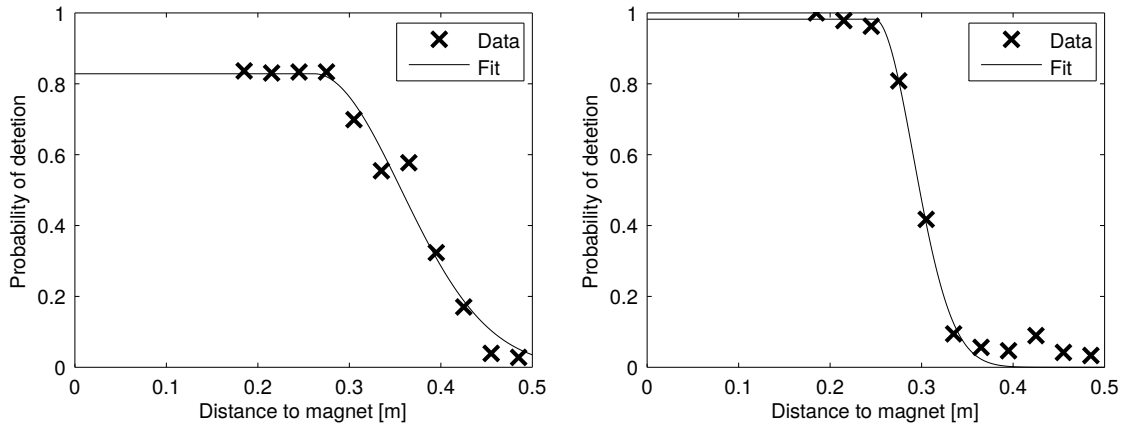


(a) MAV Kilo. (b) MAV Papa

Figure 58: Experimental results of the deviation of the magnetic field due to the addition of a magnet as a function of the true distance to the magnet for (a) MAV Kilo and (b) MAV Papa.

Figure 58. It is evident that far away from the target the deviations from the nominal field are relatively small, though with a few clutter detections. Near to the targets there are significant deviations in the magnetic field, with an approximate detection radius of 0.5 m. We have colored the data points according to whether they are inliers or outliers, using a threshold on the deviation in the field strength. For Kilo, this threshold is $\epsilon_m = 0.55$ and for Papa $\epsilon_m = 0.03$.

We use this data to characterize the detection and clutter models for the MAD sensor. The probability of a false positive is computed using the ratio of detections to all measurements outside of the sensing radius. To compute the detection statistics, we divide the distance from the robot to the target into bins (of width 3 cm) and look at the detection rate within each bin, using the thresholds from above to determine true versus missed detections. Figure 59 shows this experimental data and the best fit detection models. For both robots, the detection rate is relatively high and constant when the distance is small and falls sharply



(a) MAV Kilo (b) MAV Papa

Figure 59: Experimentally determined MAD sensor detection models used for target detection and localization.

Table 4: Best fit MAD sensor parameters.

Parameter	p_{fn}	R_0 [m]	σ_R [m]	$R_{\max}$ [m]	p_{fp}
Kilo	0.1717	0.2616	0.0948	0.5	0.0032
Papa	0.0177	0.2485	0.0425	0.5	0.0138

towards zero as the distance increases. Given this, we model the probability of detection as

$$p_d(\mathbf{x} \mid \mathbf{q}) = \begin{cases} 1 - p_{\text{fn}} & |\mathbf{x} - \mathbf{q}| < R_0, \\ (1 - p_{\text{fn}}) \exp\left(-\frac{(|\mathbf{x}-\mathbf{q}|-R_0)^2}{2\sigma_R^2}\right) & R_0 \leq |\mathbf{x} - \mathbf{q}| \leq R_{\max}, \\ 0 & R_{\max} < |\mathbf{x} - \mathbf{q}|. \end{cases} \tag{A.1}$$

Here, p_{fn} is the probability of a false negative, R_0 is the radius inside which the probability of detection is constant, σ_R is the rate at which the probability of detection drops off with distance, and $R_{\max}$ is the maximum detection range of the sensor. To find the best fit parameters, we perform a brute-force search over a range of the parameter space, selecting the model with the minimum sum-of-squares error between the data points the model. p_{fp} is the probability of a false positive detection and is found by counting the empirical fraction of detections when the magnet was further than the maximum sensing range. Table 4 lists the best fit parameters for the two MAV platforms.

Appendix B

Bearing-Only Sensor Characterization

We conduct experiments using a small team of ground robots (Scarabs), pictured in Figure 60. The Scarabs are differential drive robots with an onboard computer with an Intel i5 processor and 8 GB of RAM, running Ubuntu 12.04. They are equipped with a Hokuyo UTM-30LX laser scanner, used for self-localization and for target detection. The robots communicate with a central computer, a laptop with an Intel i7 processor and 16 GB of RAM, running ROS on Ubuntu 12.04, via an 802.11n network. The team explores in an indoor hallway, shown in Figure 61, seeking the reflective objects pictured with the robot in Figure 60.

We converted a Hokuyo into a bearing-only sensor, which may be thought of as a proxy to a camera. This simple sensor performs better than a camera in that avoids common problems with visual sensors such as variable lighting conditions and distortions. The objects for which the robots search strips of 3M 7610 reflective tape attached to PVC pipes with an outer diameter of 1.625 in. The tape provides high intensity returns to the laser scanner, allowing us to pick out objects from the background environment. However, there is no way to uniquely identify individual objects, making this the ideal setting to use the PHD filter.

The hallway features a variety of building materials such as drywall, wooden doors, painted metal (door frames), glass (office windows), and bare metal (chair legs, access panels, and drywall corner protectors, like that in the right side of Figure 60). The reflective

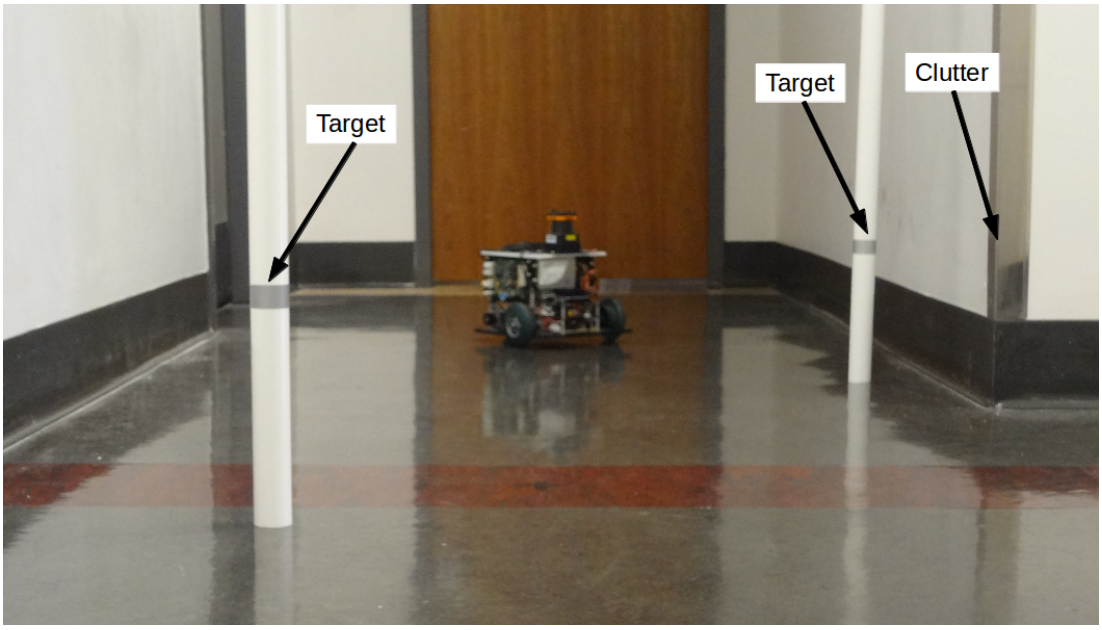


Figure 60: A Scarab robot with two targets in the experimental environment.

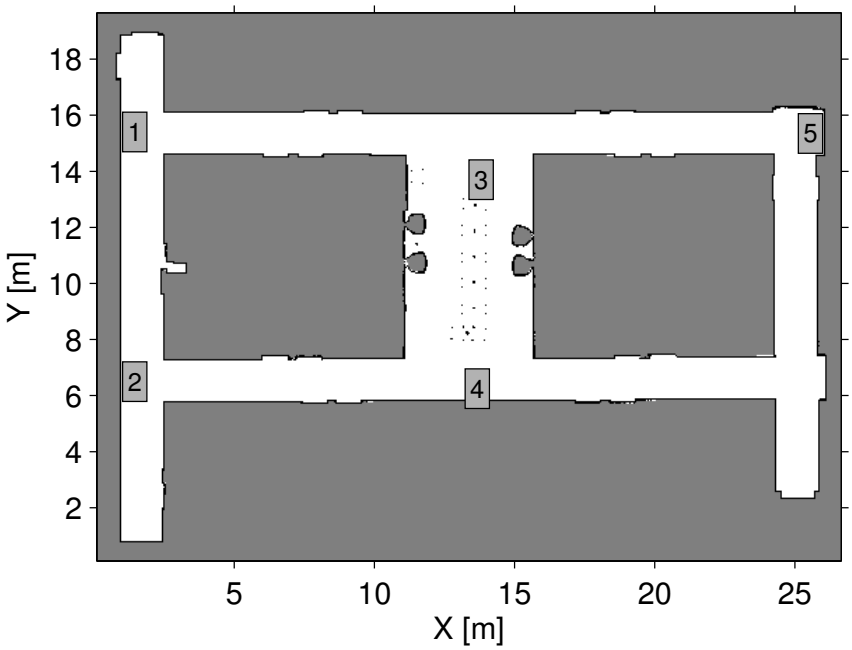


Figure 61: A floorplan of the environment used in the hardware experiments. Different starting locations for the robots are labeled in the map.

properties of the environment greatly vary for different building materials and for different angles of incidence of the laser. The intensity of bare metal and glass surfaces at low angles of incidence is similar to that of the reflective tape. Figure 63 shows an example laser scan from the environment, with the robot at the origin and oriented along the x-axis. The targets show up as clear, high intensity sections in the laser scan, though the intensity decreases with distance to the robot (objects 1–5 are placed 1–5 m from the robot, respectively).

Motivated by this, we select a threshold on the laser intensity of 11000 (shown as a black dotted line in the plots) to be able to reliably detect objects within a 5 m range of the robot. At low angles of incidence, bare metal and glass have laser returns of similarly intensity to the true objects, creating clutter measurements. Note that glass has an extremely narrow band of angles of incidence ($\approx 0.02^\circ$) that result in high intensity returns, while the metal has a wider range ($\approx 2.5^\circ$). If we were to eliminate the clutter detections, then the effective sensing range of the robots would only be less than 2 m, significantly decreasing the utility of the system.

To turn a laser scan into a set of bearing measurements, we first prune the points based on the laser intensity threshold, retaining only those with sufficiently high intensity returns. The points are clustered spatially using the range and bearing information, with each cluster having a maximum diameter d_t. The range data is otherwise discarded. The bearings to each of the resulting clusters form a measurement set Z.

A team of three robots drove around the environment for 20 min collecting measurements of 15 targets at known positions. To move around the environment, the robots generated actions over length scales ranging from 1 m to 20 m, as shown in Figure 62, and randomly selected actions from the candidate set. This allowed the robots to cover the environment much more effectively than a pure random walk. The collected data set consists of 1959 measurement sets (*i.e.*, laser scans) containing 2630 individual bearing measurements.

We now develop the detection, measurement, and clutter models necessary to utilize the PHD filter using the robot, sensor, and targets from the previous section.

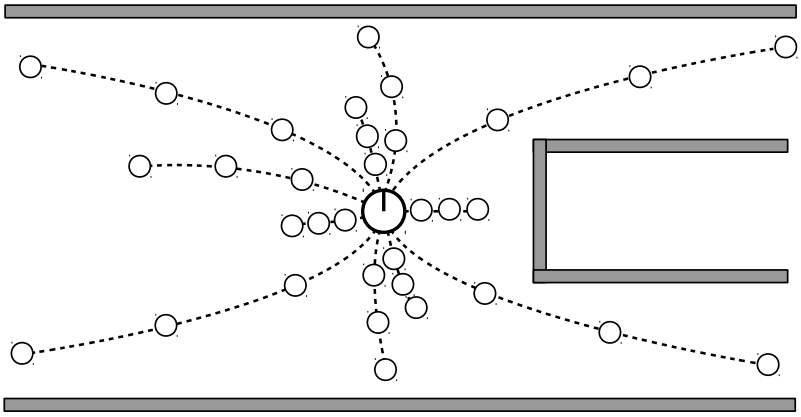
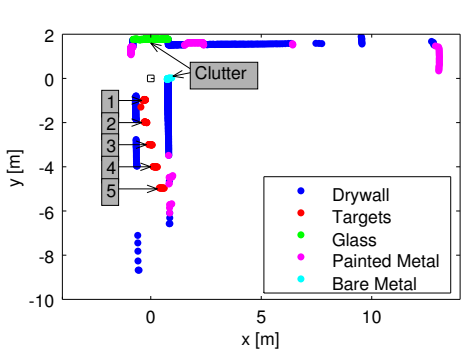
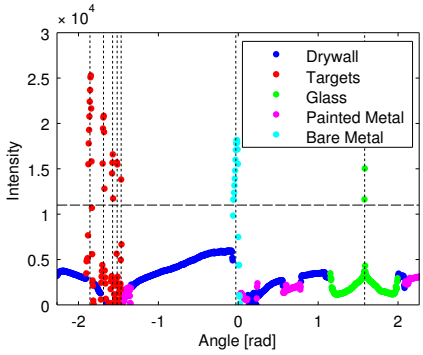


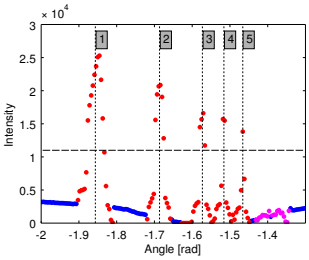
Figure 62: Example action set with a horizon of $T = 3$ steps and three length scales. Each action is a sequence of T poses at which the robot will take a measurement, denoted by the hollow circles.



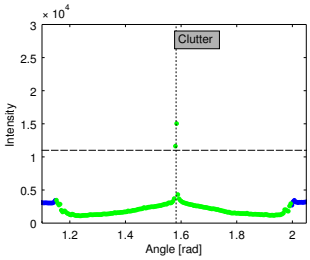
(a) XY laser scan



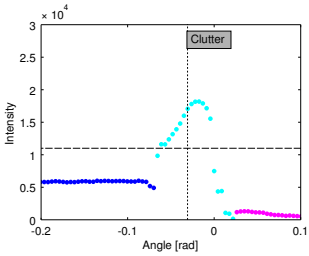
(b) Intensity plot and measurement set



(c) Targets

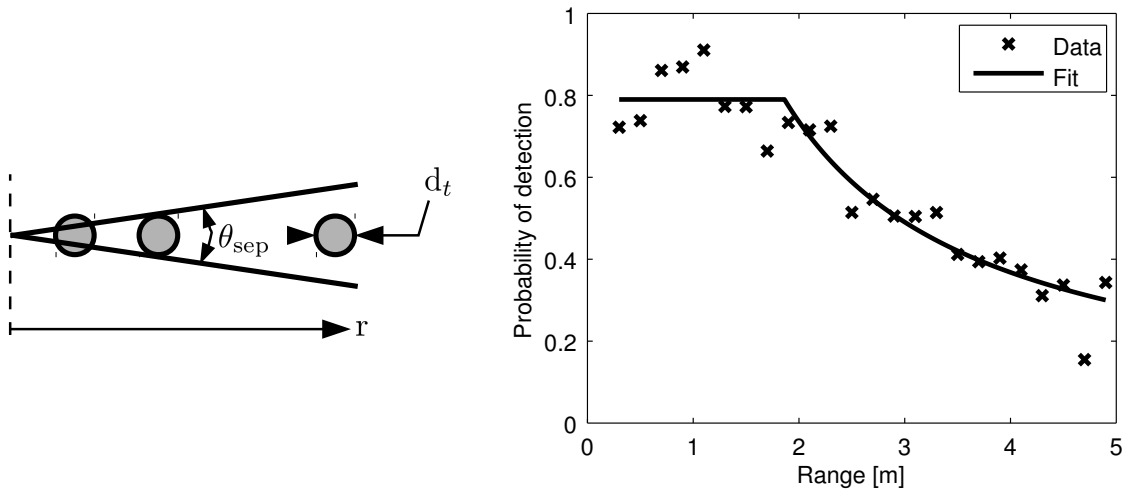


(d) Glass



(e) Metal

Figure 63: An example laser scan from the office environment. (a) Shows the XY scan labeled according to the building material. Objects 1–5 are placed 1–5 m from the robot and the sources of clutter measurements are also labeled. (b) Shows the corresponding intensity plot with material labels and the resulting measurement set. (c)-(e) Show insets of specific objects of interest within the scan and the resulting measurements.



(a) Detection diagram

(b) Detection model

Figure 64: (a) A pictogram of the laser detection model, where d_t is the diameter of the target, θ_{sep} is the angular separation between beams, and r is the range. (b) Best fit detection model to 1588 true detections, from 1959 measurement sets, and 1007 false negative detections.

B.1 Detection Model

The detection model can be determined using simple geometric reasoning due to the nature of the laser scanner, as Figure 64a shows. Each beam in a laser scan intersects a target that is within $d_t/2$ of the beam. The arc length between two beams at a range r is $r\theta_{\text{sep}}$, and the covered space is d_t. Using the small angle approximation for tangent, the probability of detection is

$$p_d(x;q) = (1-p_{\text{fn}}) \min\left(1, \frac{d_t}{r(x,q)\theta_{\text{sep}}}\right) \mathbf{1}\left(b(x,q) \in [b_{\min}, b_{\max}]\right) \mathbf{1}\left(r(x,q) \in [0, r_{\max}]\right) \quad \text{(B.1)}$$

where $r(x,q)$ and $b(x,q)$ are the range and bearing to the target in the local sensor frame, p_{fn} is the probability of a false negative, and $\mathbf{1}(\cdot)$ is an indicator function. The bearing is limited to fall within $[b_{\min}, b_{\max}]$ and the range to be less than some maximum value $r_{\max}$ (here due to the intensity threshold on the laser and the reflectivity of the targets).

To find the optimal parameter values, we take the collected data and determine which measurements originate from true objects. A measurement is labeled as a true detection if

it is within $3\,\sigma$ of the true bearing to the target and within the field of view of the robot, given its current pose. Since the PHD filter assumes that each target creates at most one measurement per scan, once a measurement-to-target association is made, that target is no longer fit to any other measurements in a measurement set. If no measurement is associated to a target, the target is labeled as a false negative detection. The collected data contains 1588 true detections and there are 1007 false negative detections.

We bin this data as a function of the true range to the target in 0.2 m increments, computing the probability of detecting a target within each range bin. We search over p_{fn} (from 0 to 1 in steps of 0.005) and d_t (from 0 in to 2 in in steps of 0.01 in), computing the sum-of-squares error between the data and the parameterized model. We find the effective target diameter d_t, since the intensity is below the cutoff threshold at extremely high angles of incidence to the reflective tape. Figure 64b shows the best fit model, with $p_{\text{fn}} = 0.210$ and $d_t = 1.28$ in. These parameters are reasonable, with the effective target diameter being 78.8% of the true target diameter. This corresponds to a maximum angle of incidence of 52.0°, which is orders of magnitude larger for the targets than for glass or metal.

B.2 Measurement Model

The sensor returns a bearing measurement to each detected target. We assume that bearing measurements are corrupted by zero-mean Gaussian noise with covariance σ, which is independent of the robot pose and of the range and bearing to the target. In other words,

$$g(z \mid x; q) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(- \frac{\big(z - b(x,q)\big)^2}{2\sigma^2} \right), \tag{B.2}$$

where $b(x, q)$ is the bearing of the target in the sensor frame.

During the runs, the robots occasionally experience significant errors in localization due to occlusions by transient objects, long feature-poor hallways, and displaced semi-static objects, *e.g.*, chairs. Since no ground-truth localization data is available in the experimental environment, we fit the noise parameter σ by searching over a range of possible values. Note that the value of σ affects the target-to-measurement association, and thus the detection

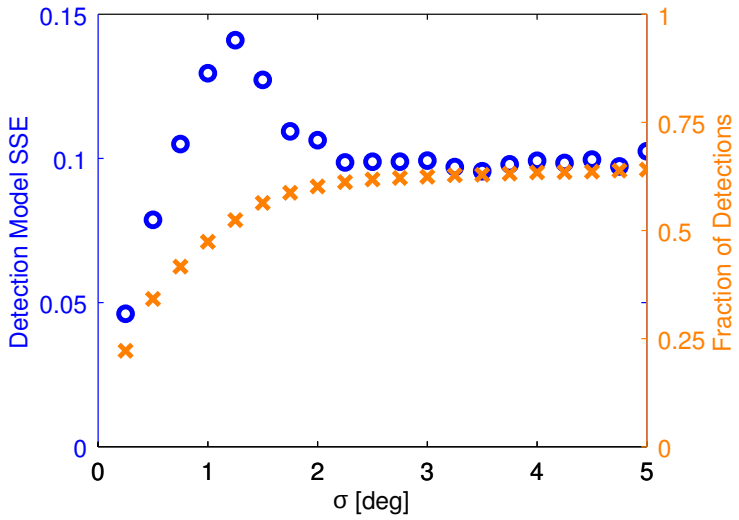


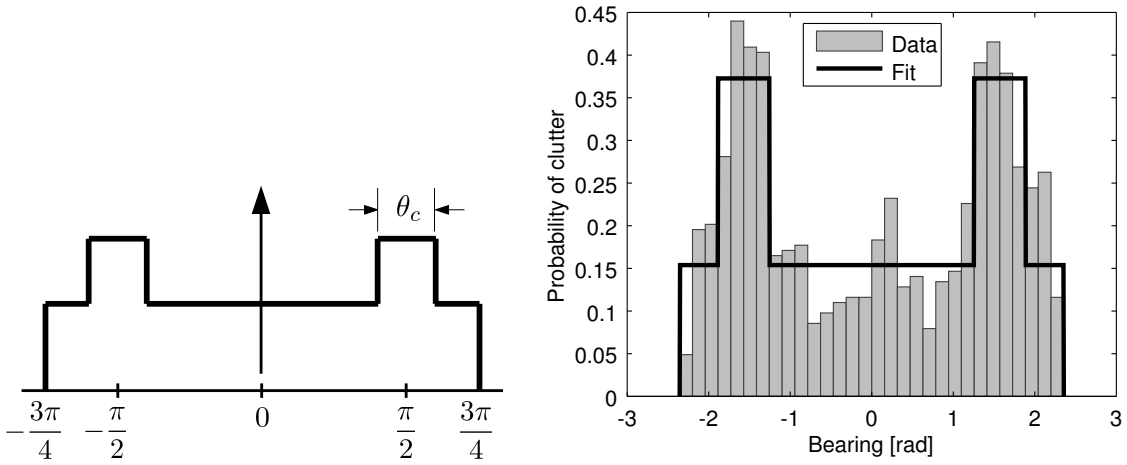
Figure 65: Detection model sum of squares error (SSE) (blue circles) and the fraction of measurements that were classified as detections (orange exes) as a function of the measurement noise parameter.

and clutter model parameters.

Figure 65 shows that the choice of the parameter σ creates a frontier for the detection model SSE and the fraction of measurements classified as detections. With very low values of σ, there are few inliers, *i.e.*, labeled detections, so the model fits well but is not meaningful. The error in the fit increases until around $\sigma = 1.25°$, when it begins to decrease. From this point, the model fits increasingly well, though after a point the decrease is due to overfitting the data and there is a clear "knee" in the data. This is also the point where the fraction of inliers levels out. We select $\sigma = 2.25°$ as the best fit measurement noise parameter, and use this to fit the detection and clutter models.

B.3 Clutter Model

Clutter (*i.e.*, false positive) measurements arise due to reflective surfaces within the environment, such as glass and bare metal, only at low angles of incidence. Since these materials are mostly found on walls, which are to the side of the robot when it is driving down a hallway, there will be a higher rate of clutter detections near $\pm\frac{\pi}{2}$ rad in the laser scan. For objects such as table and chair legs there is no clear relationship between the relative pose



(a) Clutter model (b) Clutter distribution

Figure 66: (a) A pictogram of the clutter model, where θ_c is the width of the clutter peaks centered at $\pm\frac{\pi}{2}$, and the bearing falls within the range $[-\frac{3\pi}{4}, \frac{3\pi}{4}]$. (b) Best fit clutter probability density function, using 1010 clutter measurements from 1959 measurement sets.

of the object and robot, so we assume that such detections occur uniformly across the field of view of the sensor. This leads to a clutter model of the form shown in Figure 66a.

Let θ_c be the width of the clutter peaks centered at $\pm\frac{\pi}{2}$ and let p_u be the probability that a clutter measurements was generated from a target in the uniform component of the clutter model. The clutter model is

$$c(z) = \frac{p_u \mu}{b_{\max} - b_{\min}} \mathbf{1}\left(b \in [b_{\min}, b_{\max}]\right) + \frac{(1-p_u)\mu}{2\theta_c} \mathbf{1}\left(\left||b| - \pi/2\right| \leq \theta_c/2\right), \tag{B.3}$$

where μ is the expected number of clutter measurements per scan. The clutter cardinality, m, is assumed to follow a Poisson distribution with mean μ [67].

All measurements that are not associated to a target, as described in Sec. B.1, are considered to be clutter measurements. We bin the bearings of these clutter measurements in $\frac{\pi}{20}$ increments to create a piecewise-constant distribution. We perform a search over θ_c (from 0 rad to $\frac{\pi}{2}$ rad in steps of $\frac{\pi}{400}$ rad) and p_u (from 0 to 1 in steps of 0.005) to find the best fit parameters (using the sum-of-squares error) to the data, with Figure 66b showing the best fit model, with $\theta_c = 0.200\pi$ rad and $p_u = 0.725$. The number of clutter measurements

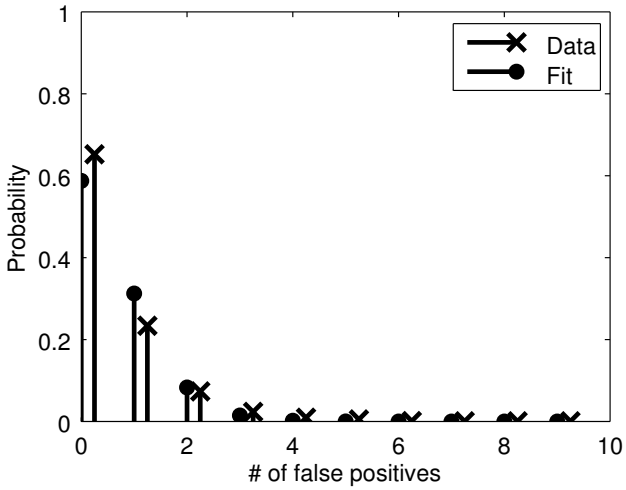


Figure 67: Best fit model for the clutter cardinality, using 1010 clutter measurements from 1959 measurement sets.

per scan is used to fit the clutter cardinality parameter μ, with Figure 67 showing the best fit value, $\mu = 0.5319$.

B.4 Analysis

The specific values of the sensor parameters depend upon the specific robot, sensor, targets, and environment. For example, the peaks in the clutter model near $\pm\frac{\pi}{2}$ arise due to the geometry and appearance of the environment. In a more open setting, or with a more limited field of view sensor, we would not expect to see these peaks in the clutter distribution. Or in an environment with more glass walls, we would expect the number of clutter detections to be higher and the peaks to be more pronounced.

The detection statistics also depend highly upon our particular experimental setup. The strip of reflective tape is only 1 in tall and the sensor is planar, so a bump in the floor of only 2.5 mm would cause the robot to pitch sufficiently to fail to detect a target that is 1 m away. Any small bumps in the linoleum flooring, particularly at transitions to carpeting, cause the robot to experience false negative detections. Additional false negatives may occur due to occlusions from transient objects, *e.g.*, passing people and other robots, and semi-static objects such as chairs.

Bibliography

[1] Robot Operating System. `http://www.ros.org/wiki/`, March 2013.

[2] Martin Adams, Ba-Ngu Vo, and Ronald Mahler, editors. *Advances in Probabilistic Modeling: Applications of Stochastic Geometry*, volume 21(2) of *IEEE Robotics and Automation Magazine*, June 2014. IEEE.

[3] Nikolay Atanasov, Jerome Le Ny, Kostas Daniilidis, and George Pappas. Information Acquisition with Sensing Robots: Algorithms and Error Bounds. In *IEEE International Conference on Robotics and Automation*, pages 6447–6454, 2014. URL `http://dx.doi.org/10.1109/ICRA.2014.6907811`.

[4] Nikolay Atanasov, Menglong Zhu, Kostas Daniilidis, and George J Pappas. Semantic Localization Via the Matrix Permanent. In *Robotics: Science and Systems*, 2014.

[5] Nikolay Atanasov, Jerome Le Ny, Kostas Daniilidis, and George Pappas. Decentralized Active Information Acquisition: Theory and Application to Multi-Robot SLAM. In *IEEE International Conference on Robotics and Automation*, 2015.

[6] Frédéric Bourgault, Alexei Makarenko, Stefan Williams, Ben Grocholsky, and Hugh Durrant-Whyte. Information Based Adaptive Robotic Exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 540–545, 2002.

[7] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *Integer programming and combinatorial optimization*, pages 182–196. Springer, 2007.

[8] Stefano Carpin, Derek Burch, and Timothy Chung. Searching for Multiple Targets Using Probabilistic Quadtrees. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4536–4543. IEEE, 2011.

[9] Henry Carrillo, Ian Reid, and José A. Castellanos. On the Comparison of Uncertainty Criteria for Active SLAM. In *IEEE International Conference on Robotics and Automation*, pages 2080–2087, 2012.

[10] Henry Carrillo, Philip Dames, Vijay Kumar, and José Castellanos. Autonomous Robotic Exploration Using Occupancy Grid Maps and Graph SLAM Based on Shannon and Rényi Entropy. In *IEEE International Conference on Robotics and Automation*, pages 487–494. IEEE, May 2015.

[11] Benjamin Charrow, Vijay Kumar, and Nathan Michael. Approximate Representations for Multi-Robot Control Policies that Maximize Mutual Information. *Robotics: Science and Systems*, 2013.

[12] Benjamin Charrow, Nathan Michael, and Vijay Kumar. Active Control Strategies for Discovering and Localizing Devices with Range-Only Sensors. In *Workshop on the Algorithmic Foundations of Robotics*, 2014.

[13] Han-Lim Choi and Jonathan P How. Continuous Trajectory Planning of Mobile Sensors for Informative Forecasting. *Automatica*, 46(8):1266–1275, August 2010.

[14] Han-Lim Choi and Jonathan P How. Coordinated Targeting of Mobile Sensor Networks for Ensemble Forecast Improvement. *IEEE Sensors Journal*, 11(3):621–633, 2011.

[15] Han-Lim Choi and Jonathan P How. Efficient Targeting of Sensor Networks for Large-Scale Systems. *IEEE Transactions on Control Systems Technology*, 19(6):1569–1577, November 2011.

[16] Han-Lim Choi, Jonathan P How, and P. I. Barton. An Outer-Approximation Algorithm for Generalized Maximum Entropy Sampling. In *Proceedings of the American Control Conference*, pages 1818–1823, Seattle, WA, USA, June 2008.

[17] Timothy H Chung, Geoffrey A Hollinger, and Volkan Isler. Search and pursuit-evasion in mobile robotics. *Autonomous robots*, 31(4):299–316, 2011.

[18] David Cole. *A Cooperative UAS Architecture for Information-Theoretic Search and Track*. PhD thesis, University of Sydney, 2009.

[19] Randy A Cortez, Herbert G Tanner, Ron Lumia, and Chaouki T Abdallah. Information Surfing for Radiation Map Building. *International Journal of Robotics and Automation*, 26(1):4, 2011.

[20] Thomas Cover and Joy Thomas. *Elements of Information Theory*. John Wiley & Sons, 2nd edition, 2012.

[21] Daryl J Daley and David Vere-Jones. *An Introduction to the Theory of Point Processes*, volume 2. Springer, 2003.

[22] Philip Dames and Vijay Kumar. Cooperative Multi-Target Localization With Noisy Sensors. In *IEEE International Conference on Robotics and Automation*, pages 1877–1883, May 2013.

[23] Philip Dames and Vijay Kumar. Experimental Characterization of a Bearing-only Sensor for Use With the PHD Filter, 2015. Available at: arXiv:1502.04661 [cs:RO].

[24] Philip Dames and Vijay Kumar. Multi-Robot Detection and Localization of Targets Using Probability Hypothesis Density Functions. In *IEEE International Conference on Automation Science and Engineering*, May 2015. Accepted.

[25] Philip Dames and Vijay Kumar. Autonomous Localization of an Unknown Number of Targets without Data Association Using Teams of Mobile Sensors. 2015. To appear.

[26] Philip Dames, Mac Schwager, Vijay Kumar, and Daniela Rus. A Decentralized Control Policy for Adaptive Information Gathering in Hazardous Environments. In *IEEE International Conference on Decision and Control*, pages 2807–2813, December 2012.

[27] Philip Dames, Dinesh Thakur, Mac Schwager, and Vijay Kumar. Playing Fetch With Your Robot: The Ability of Robots to Locate and Interact with Objects. 21(2):46–52, 2014.

[28] Philip Dames, Pratap Tokekar, and Vijay Kumar. Detecting, Localizing, and Tracking an Unknown Number of Moving Targets Using a Team of Mobile Robots. In *International Symposium on Robotics Research (ISRR)*, September 2015. Accepted.

[29] Aveek Das, Dinesh Thakur, James Keller, Sujit Kuthirummal, Zsolt Kira, and Mihail Pivtoraiko. R-MASTIF: Robotic Mobile Autonomous System for Threat Interrogation and Object Fetch. In *IS&T/SPIE Electronic Imaging*, pages 86620O–86620O. International Society for Optics and Photonics, 2013.

[30] Francesco. Delle Fave, Alex Rogers, Zhe Xu, Salah Sukkarieh, and Nicholas Jennings. Deploying the Max-Sum Algorithm for Decentralised Coordination and Task Allocation of Unmanned Aerial Vehicles for Live Aerial Imagery Collection. In *IEEE International Conference on Robotics and Automation*, pages 469–476. IEEE, 2012.

[31] MWM Gamini Dissanayake, Paul Newman, Steve Clark, Hugh F Durrant-Whyte, and Michael Csorba. A Solution to the Simultaneous Localization and Map Building (SLAM) Problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001.

[32] Alberto Elfes. Using Occupancy Grids for Mobile Robot Perception and Navigation. *Computer*, 22(6):46–57, 1989.

[33] Ozgur Erdinc, Peter Willett, and Yaakov Bar-Shalom. The Bin-Occupancy Filter and its Connection to the PHD Filters. *IEEE Transactions on Signal Processing*, 57(11): 4232–4246, 2009.

[34] Eric W Frew and Stephen M Rock. Trajectory generation for constant velocity target motion estimation using monocular vision. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 3479–3484, 2003.

[35] Kevin Fu. RFID-Scale Devices in Concrete, January 2014. URL `https://spqr.eecs.umich.edu/moo/apps/concrete/`.

[36] Tomonari Furukawa, Frederic Bourgault, Benjamin Lavis, and Hugh F Durrant-Whyte. Recursive Bayesian search-and-tracking using coordinated UAVs for lost targets. In *IEEE International Conference on Robotics and Automation*, pages 2521–2526, 2006.

[37] Nicholas R Gans, Guoqiang Hu, Kaushik Nagarajan, and Warren E Dixon. Keeping multiple moving targets in the field of view of a mobile camera. *IEEE Transactions on Robotics*, 27(4):822–828, 2011.

[38] Brian Gerkey. gmapping, July 2014. URL `http://wiki.ros.org/gmapping`.

[39] Helmut Grabner, Thuy Thi Nguyen, Barbara Gruber, and Horst Bischof. On-line boosting-based car detection from aerial images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 63(3):382–396, 2008.

[40] S. Grime and Hugh Durrant-Whyte. Data Fusion in Decentralized Sensor Networks. *Control engineering practice*, 2(5):849–863, 1994.

[41] Ben Grocholsky. *Information-Theoretic Control of Multiple Sensor Platforms*. PhD thesis, University of Sydney, 2002.

[42] Ben Grocholsky, James Keller, Vijay Kumar, and George Pappas. Cooperative air and ground surveillance. *IEEE Robotics and Automation Magazine*, 13(3):16–25, 2006.

[43] Erico Guizzo. Robots With Their Heads in the Clouds. *IEEE Spectrum*, 48(3):16–18, 2011.

[44] Erico Guizzo. Cloud Robotics: Connected to the Cloud, Robots Get Smarter. *IEEE Spectrum*, 2011. URL `http://spectrum.ieee.org/automaton/robotics/robotics-software/cloud-robotics`.

[45] Gabriel Hoffmann and Claire Tomlin. Mobile Sensor Network Control Using Mutual Information Methods and Particle Filters. *IEEE Transactions on Automatic Control*, pages 1–16, 2010.

[46] Geoffrey A Hollinger, Joseph Djugash, and Sanjiv Singh. Target tracking without line of sight using range from radio. *Autonomous Robots*, 32(1):1–14, 2012.

[47] Geoffrey A. Hollinger, Brendan Englot, Franz S. Hover, Urbashi Mitra, and Gaurav S. Sukhatme. Active Planning for Underwater Inspection and the Benefit of Adaptivity. *The International Journal of Robotics Research*, 32(1):3–18, November 2012.

[48] Ali Jadbabaie. *Receding Horizon Control of Nonlinear Systems: A Control Lyapunov Function Approach*. PhD thesis, California Institute of Technology, 2000.

[49] Joshua Joseph, Finale Doshi-Velez, Albert S Huang, and Nicholas Roy. A bayesian nonparametric approach to modeling motion patterns. *Autonomous Robots*, 31(4): 383–400, 2011.

[50] Brian Julian, Michael Angermann, Mac Schwager, and Daniela Rus. A Scalable Information Theoretic Approach to Distributed Robot Coordination. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[51] Brian Julian, Michael Angermann, Mac Schwager, and Daniela Rus. Distributed Robotic Sensor Networks: An Information-Theoretic Approach. *The International Journal of Robotics Research*, 31(10):1134–1154, August 2012.

[52] Brian Julian, Sertac Karaman, and Daniela Rus. On Mutual Information-Based Control of Range Sensing Robots for Mapping Applications. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5156–5163. IEEE, November 2013.

[53] Rudolph Emil Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Fluids Engineering*, 82(1):35–45, 1960.

[54] Chang-Young Kim, Dezhen Song, Yiliang Xu, Jingang Yi, and Xinyu Wu. Cooperative search of multiple unknown transient radio sources using multiple paired mobile robots. *IEEE Transactions on Robotics*, 30(5):1161–1173, Oct 2014. ISSN 1552-3098. doi: 10.1109/TRO.2014.2333097.

[55] David Kotz and Tristan Henderson. CRAWDAD: A community resource for archiving wireless data at dartmouth. *IEEE Pervasive Computing*, 4(4):12–14, 2005.

[56] Andreas Krause and Carlos Guestrin. Near-optimal Observation Selection Using Submodular Functions. In *Proceedings of 22nd Conference on Artificial Intelligence (AAAI)*, Vancouver, Canada, July 2007.

[57] Andreas Krause and Carlos E. Guestrin. Near-optimal Nonmyopic Value of Information in Graphical Models. *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, 2005. URL `http://arxiv.org/abs/1207.1394`.

[58] Andreas Krause, Carlos Guestrin, Anupam Gupta, and Jon Kleinberg. Near-optimal Sensor Placements: Maximizing Information while Minimizing Communication Cost. In *In Proceedings of Information Processing in Sensor Networks (IPSN)*, Nashville, TN, April 19-21 2006.

[59] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. *Journal of Machine Learning Research*, 9:235–284, 2008.

[60] Chris Kreucher, Keith Kastella, and Alfred O Hero III. Sensor management using an active sensing approach. *Signal Processing*, 85(3):607–624, 2005.

[61] X Rong Li and Vesselin P Jilkov. Survey of maneuvering target tracking. part i. dynamic models. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4): 1333–1364, 2003.

[62] X Rong Li and Vesselin P Jilkov. Survey of maneuvering target tracking. part v. multiple-model methods. *IEEE Transactions on Aerospace and Electronic Systems*, 41(4):1255–1321, 2005.

[63] Xu Li, Rongxing Lu, Xiaohui Liang, Xuemin Shen, Jiming Chen, and Xiaodong Lin. Smart community: an internet of things application. *IEEE Communications Magazine*, 49(11):68–75, 2011.

[64] Andrew Logothetis, Alf Isaksson, and Robin J Evans. An information theoretic approach to observer path design for bearings-only tracking. In *IEEE International Conference on Decision and Control*, volume 4, pages 3132–3137, 1997.

[65] Christian Lundquist, Lars Hammarstrand, and Fredrik Gustafsson. Road Intensity Based Mapping Using Radar Measurements With a Probability Hypothesis Density Filter. *IEEE Transactions on Signal Processing*, 59(4):1397–1408, 2011. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5677613`.

[66] Kevim M Lynch, Ira B Schwartz, Peng Yang, and Randy A Freeman. Decentralized Environmental Modeling by Mobile Sensor Networks. *IEEE Transactions on Robotics*, 24(3):710–724, June 2008.

[67] Ronald Mahler. Multitarget Bayes Filtering via First-Order Multitarget Moments. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1152–1178, October 2003.

[68] Ronald Mahler. Objective Functions for Bayesian Control-Theoretic Sensor Management, 1: Multitarget First-Moment Approximation. In *IEEE Aerospace Conference Proceedings*, volume 4. Ieee, 2003. doi: 10.1109/AERO.2003.1235121.

[69] Ronald Mahler. PHD Filters of Higher Order in Target Number. *IEEE Transactions on Aerospace and Electronic Systems*, (4), 2007.

[70] Ronald Mahler. *Statistical Multisource-Multitarget Information Fusion*, volume 685. Artech House Boston, 2007.

[71] James Manyika, Michael Chui, Jacques Bughin, Richard Dobbs, Peter Bisson, and Alex Marrs. *Disruptive technologies: Advances that will transform life, business, and the global economy*, volume 180. McKinsey Global Institute San Francisco, CA, 2013.

[72] David Q Mayne and Hannah Michalska. Receding Horizon Control of Nonlinear Systems. *IEEE Transactions on Automatic Control*, 35(7):814–824, 1990.

[73] David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained Model Predictive Control: Stability and Optimality. *Automatica*, 36(6): 789–814, 2000.

[74] Nathan Michael, Jonathan Fink, Savvas Loizou, and Vijay Kumar. Architecture, Abstractions, and Algorithms for Controlling Large Teams of Robots: Experimental Testbed and Results. *International Journal of Robotics Research*, pages 409–419, 2011.

[75] Diluka Moratuwage, Ba-Ngu Vo, and Danwei Wang. Collaborative Multi-Vehicle SLAM with Moving Object Tracking. *IEEE International Conference on Robotics and Automation*, pages 5702–5708, May 2013. doi: 10.1109/ICRA.2013.6631397.

[76] John Mullane, Ba-Ngu Vo, Martin D Adams, and Ba-Tuong Vo. A Random-Finite-Set Approach to Bayesian SLAM. *IEEE Trans. on Robotics*, 27(2):268–282, 2011.

[77] John Mullane, Ba-Ngu Vo, Martin D Adams, and Ba-Tuong Vo. Random Finite Sets for Robot Mapping and SLAM. *Springer Tracts in Advanced Robotics*, 2011.

[78] Daniel P Palomar and Sergio Verdú. Gradient of Mutual Information in Linear Vector Gaussian Channels. *IEEE Transactions on Information Theory*, 52(1):141–154, January 2006.

[79] Daniel P Palomar and Sergio Verdú. Representation of Mutual Information Via Input Estimates. *IEEE Transactions on Information Theory*, 53(2):453–470, February 2007.

[80] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. CRAWDAD data set epfl/mobility (v. 2009-02-24). Downloaded from http://crawdad.org/epfl/mobility/, February 2009.

[81] Friedrich Pukelsheim. *Optimal Design of Experiments*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 2006.

[82] GW Pulford. Taxonomy of Multiple Target Tracking Methods. In *IEEE Procs. on Radar, Sonar and Navigation*, volume 152, pages 291–304. IET, 2005.

[83] Carl Rasmussen and Chris Williams. Gaussian processes for machine learning. *Gaussian Processes for Machine Learning*, 2006.

[84] Stephan Reuter, Ba-Tuong Vo, Ba-Ngu Vo, and Klaus Dietmayer. The labeled multi-bernoulli filter. *IEEE Transactions on Signal Processing*, 62(12):3246–3260, 2014.

[85] Branko Ristic and Ba-Ngu Vo. Sensor Control for Multi-Object State-Space Estimation Using Random Finite Sets. *Automatica*, 46(11):1812–1818, November 2010.

[86] Branko Ristic, Daniel Clark, and Ba-Ngu Vo. Improved smc implementation of the phd filter. In *IEEE Conference on Information Fusion (FUSION)*, pages 1–8. IEEE, 2010.

[87] Branko Ristic, Ba-Ngu Vo, and Daniel Clark. A Note on the Reward Function for PHD Filters with Sensor Control. *IEEE Transactions on Aerospace and Electronic Systems*, 47(2):1521–1529, 2011.

[88] Anthony Rowe, Mario Berges, Gaurav Bhatia, Ethan Goldman, Raj Rajkumar, James H. Garrett, Jose M. F. Moura, and Lucio Soibelman. Sensor Andrew: Large-scale Campus-Wide Sensing and Actuation. *IBM Journal of Research and Development*, 55(1.2):6:1–6:14, January 2011.

[89] Allison Ryan. *Information-Theoretic Control for Mobile Sensor Teams*. PhD thesis, University of California, Berkeley, 2008.

[90] Mac Schwager, Philip Dames, Daniela Rus, and Vijay Kumar. A Multi-Robot Control Policy for Information Gathering in the Presence of Unknown Hazards. In *International Symposium on Robotics Research (ISRR)*, August 2011.

[91] Claude E. Shannon. A Mathematical Theory of Communication. *Bell Systems Technical Journal*, 27:379–423, 1948.

[92] Dezhen Song, Chang-Young Kim, and Jingang Yi. Simultaneous localization of multiple unknown and transient radio sources using a mobile robot. *IEEE Transactions on Robotics*, 28(3):668–680, 2012.

[93] Jefferson R Souza, Roman Marchant, Lionel Ott, Denis F Wolf, and Fabio Ramos. Bayesian Optimisation for Active Perception and Smooth Navigation. In *IEEE International Conference on Robotics and Automation*, 2014.

[94] John R Spletzer and Camillo J Taylor. Dynamic sensor planning and control for optimally tracking targets. *The International Journal of Robotics Research*, 22(1): 7–20, 2003.

[95] Lawrence D Stone, Roy L Streit, Thomas L Corwin, and Kristine L Bell. *Bayesian Multiple Target Tracking*. Artech House, 2013.

[96] Ruben Stranders, Alessandro Farinelli, Alex Rogers, and Nicholas Jennings. Decentralised Coordination of Mobile Sensors Using the Max-Sum Algorithm. In *International Joint Conference on AI (IJCAI)*, pages 299–304, 2009.

[97] Sebastian Thrun, Woldram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, 2005.

[98] Pratap Tokekar, Elliot Branson, Joshua Vander Hook, and Volkan Isler. Tracking aquatic invaders: Autonomous robots for monitoring invasive fish. *IEEE Robotics and Automation Magazine*, 20(3):33–41, 2013.

[99] Pratap Tokekar, Volkan Isler, and Antonio Franchi. Multi-target visual tracking with aerial robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3067–3072. IEEE, 2014.

[100] Paul Viola and William M Wells III. Alignment by maximization of mutual information. *International Journal of Computer Vision*, 24(2):137–154, 1997.

[101] Ba-Ngu Vo and Wing-Kin Ma. The Gaussian Mixture Probability Hypothesis Density Filter. *IEEE Transactions on Signal Processing*, 54(11):4091–4104, November 2006.

[102] Ba-Ngu Vo, Sumeetpal Singh, and Arnaud Doucet. Sequential Monte Carlo Methods for Multi-Target Filtering With Random Finite Sets. *IEEE Transactions on Aerospace and Electronic Systems*, 41(4):1224–1245, October 2005.

[103] Ba-Tuong Vo, Ba-Ngu Vo, and Antonio Cantoni. Analytic Implementations of the Cardinalized Probability Hypothesis Density Filter. *IEEE Transactions on Signal Processing*, 55(7):3553–3567, July 2007.

[104] Ba-Tuong Vo, Ba-Ngu Vo, and Antonio Cantoni. The cardinality balanced multi-target multi-bernoulli filter and its implementations. *IEEE Transactions on Signal Processing*, 57(2):409–423, 2009.

[105] Sonia Waharte, Andrew Symington, and Niki Trigoni. Probabilistic Search With Agile UAVs. In *IEEE International Conference on Robotics and Automation*, pages 2840–2845. IEEE, 2010.

[106] Zhe Xu, Robert Fitch, James Underwood, and Salah Sukkarieh. Decentralized coordinated tracking with mixed discrete–continuous decisions. *Journal of Field Robotics*, 30(5):717–740, 2013.

[107] Tao Zhao and Ram Nevatia. Car detection in low resolution aerial images. *Image and Vision Computing*, 21(8):693–703, 2003.

[108] Ke Zhou and Stergios I Roumeliotis. Optimal motion strategies for range-only constrained multisensor target tracking. *IEEE Transactions on Robotics*, 24(5):1168–1185, 2008.

[109] Ke Zhou and Stergios I Roumeliotis. Multirobot active target tracking with combinations of relative observations. *IEEE Transactions on Robotics*, 27(4):678–695, 2011.